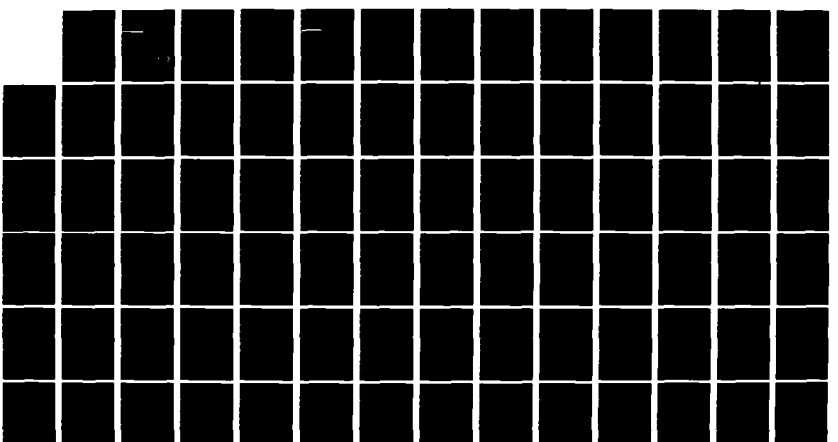
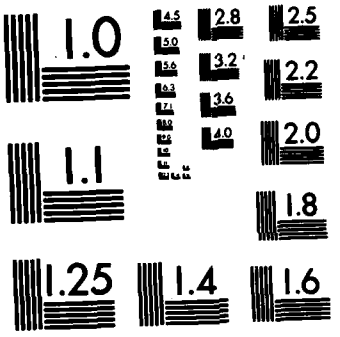
AD-A125 253 NIGEL: A SYSTEMIC GRAMMAR FOR TEXT GENERATION(U) 1/1
UNIVERSITY OF SOUTHERN CALIFORNIA MARINA DEL REY
INFORMATION SCIENCES INST W C MANN ET AL. FEB 83
UNCLASSIFIED ISI/RR-83-105 F49620-79-C-0181 . F/G 9/2 NL



END

FILMED

DTIC



ISI/RR-83-105
February 1983

12

William C. Mann
Christian M.I.M. Matthiessen

Nigel: A Systemic Grammar for Text Generation

INFORMATION SCIENCES INSTITUTE
4676 Admiralty Way/ Marina del Rey/ California 90291
(213) 822-1511
UNIVERSITY OF SOUTHERN CALIFORNIA

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ISI/RR-83-105	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Nigel: A Systemic Grammar for Text Generation		5. TYPE OF REPORT & PERIOD COVERED Research Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) William C. Mann Christian M. I. M. Matthiessen		8. CONTRACT OR GRANT NUMBER(s) F49620-79-C-0181
9. PERFORMING ORGANIZATION NAME AND ADDRESS USC/Information Sciences Institute 4676 Admiralty Way Marina del Rey, CA 90291		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Mathematical and Information Sciences Air Force Office of Scientific Research Building 410 Bolling Air Force Base Washington, D.C. 20332		12. REPORT DATE February 1983 13. NUMBER OF PAGES 82
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

This document is approved for public release and sale; distribution is unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

.....

18. SUPPLEMENTARY NOTES

The papers in this report will also appear in a forthcoming volume of the *Advances in Discourse Processes Series*, R. Freedle (ed.): *Systemic Perspectives on Discourse: Selected Theoretical Papers from the 9th International Systemic Workshop* to be published by Ablex.

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

AI, artificial intelligence, choice expert, chooser framework, discourse, grammar, inquiry semantics, rhetoric, syntax, systemic grammar, text generation, text production

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

(OVER)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

20. ABSTRACT

Programming a computer to write text which meets a prior need is a challenging research task. As part of such research, Nigel, a large programmed grammar of English, has been created in the framework of systemic linguistics begun by Halliday. In addition to specifying functions and structures of English, Nigel has a novel semantic stratum which specifies the situations in which each grammatical feature should be used.

The report consists of three papers on Nigel: an introductory overview, the script of a demonstration of its use in generation, and an exposition of how Nigel relates to the systemic framework. Although the effort to develop Nigel is significant both as computer science research and as linguistic inquiry, the outlook of the report is oriented to its linguistic significance.

ISI/RR-83-105
February 1983

William C. Mann
Christian M.I.M. Matthiessen

Nigel: A Systemic Grammar for Text Generation

INFORMATION SCIENCES INSTITUTE
UNIVERSITY OF SOUTHERN CALIFORNIA
4676 Admiralty Way/ Marina del Rey/ California 90291
(213) 822-1511

This research was supported by the Air Force Office of Scientific Research Contract No. F49620-79-C-0181. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research of the U.S. Government.

CONTENTS

OVERVIEW

The report consists of three nearly independent papers on Nigel. The first is a relatively nontechnical overview, the second is an extended example of how Nigel works, showing the coordination of syntactic and semantic parts, and the third describes the relationships between the framework implemented in Nigel and its systemic precedents. The first and third papers rely on the second for examples.

1. AN INTRODUCTION TO THE NIGEL TEXT GENERATION GRAMMAR

William C. Mann

ABSTRACT

One way to develop a functional account of how language is used would be to attempt to specify in detail how texts can be created in response to *text needs*. A text need is the earliest recognition on the part of the speaker[1] that the immediate situation is one in which he would like to produce speech. Text needs thus arise at the beginning of each turn of a dialogue, before writing personal letters or books, and generally in all kinds of situations in which language use is found. Text needs are logically prior to deciding what to say or how to say it. Research on computer text generation is one way of attempting to say how texts can be created.

Research on the text generation task has led to creation of a large systemic grammar of English, embedded in a computer program and fitted with a semantic stratum. The grammar and program, separately and jointly called Nigel, generates sentences and other units under several kinds of experimental control.

This paper gives an overview of Nigel, emphasizing the ways in which it has augmented various precedents in the systemic framework and indicating the current state of development.

1.1 A GRAMMAR FOR TEXT GENERATION--THE CHALLENGE

Among the various uses for grammars, text generation at first seems to be relatively new. The organizing goal of text generation, as a research task, is to describe how texts can be created in fulfillment of text needs. Such a description must relate texts to needs, and so must contain a functional account of the use and nature of language, a very old goal. Computational text generation research should be seen as simply a particular way to pursue that goal.

As part of a text generation research project, a grammar of English has been created and embodied in a computer program. This grammar and program, called Nigel, is intended as a component of a larger program called Penman. This paper introduces Nigel, with just enough detail about Penman to show Nigel's potential use in a text generation system.

1.1.1 The Text Generation Task as a Stimulus for Grammar Design

Text generation has been taken up only relatively recently as a computational research task [Mann 81]. It is a research area jointly included in artificial intelligence, computational linguistics and linguistics. Text generation seeks to characterize the use of natural languages by developing processes (computer programs) which can create appropriate, fluent text on demand. A

---

[1] In this report we will alternate freely between the terms speaker, writer and author, between hearer and reader, and between speech and text. This is simply partial accommodation of prevailing jargon; no differences are intended.

representative research goal would be to create a program which could write a text that serves as a commentary on a game transcript, making the events of the game understandable.[2] Because this kind of research is in a very preliminary stage, text generation serves more as a way of thinking about text and its functions than as an established collection of results.

The guiding aims in the ongoing design of the Penman text generation program are as follows:

1. To learn, in a more specific way than has previously been achieved, how appropriate text can be created in response to text needs.
2. To identify the dominant characteristics which make a text appropriate for meeting its need.
3. To develop a demonstrable capacity to create texts which meet some identifiable practical class of text needs.

Seeking to fill these goals, several different grammatical frameworks were considered. The systemic framework was chosen, and it has proven to be an entirely agreeable choice. Although it is relatively unfamiliar to many American researchers, it has a long history of work on concerns which are central to text generation. It was used by Winograd in the SHRDLU system, and more extensively by others since [Winograd 72, Davey 79, McKeown 82, McDonald 80]. A recent state of the art survey identifies the systemic framework as one of a small number of linguistic frameworks which are likely to be the basis for significant text generation programs in this decade [Mann 81].

One of the principal advantages of the systemic framework is its strong emphasis on "functional" explanations of grammatical phenomena. Each distinct kind of grammatical entity is associated with an expression of what it does for the speaker, so that the grammar indicates not only what is possible but why it would be used. Another is its emphasis on principled, justified descriptions of the ***choices*** which the grammar offers, all of its optionality. Both of these support text generation programming significantly.

This report was originally prepared for inclusion in a book on current work in systemic linguistics, so familiarity with the approach has been presumed. Basic references on the systemic framework include [Berry 75, Berry 77, Halliday 76a, Halliday 76b, Hudson 76, Halliday 81, de Joia 80, Fawcett 80].[3] For these and other reasons the systemic framework was chosen for Nigel.

In adopting the systemic framework, we have followed the preferences and values which motivate the work, and have used systemic notation. Within these guidelines we have usually followed (or been led by) Halliday.

---

[2] This was accomplished in work by Anthony Davey [Davey 79]; [McKeown 82] is a comparable more recent study.

[3] This work would not have been possible without the active participation of Christian Matthiessen, and the participation and past contributions of Michael Halliday and other systemicists.

1.1.2 Design Goals for the Grammar

Three kinds of goals have guided the work of creating Nigel.

1. To specify in total detail how ***the systemic framework*** can generate syntactic units, using the computer as the medium of experimentation.
2. To develop a ***grammar of English*** which is a good representative of the systemic framework and useful for demonstrating text generation on a particular task.
3. To specify how the grammar can be ***regulated effectively by the prevailing text need*** in its generation activity.

Each of these has led to a different kind of activity in developing Nigel and a different kind of specification in the resulting program, as described below.

The three design goals above have not all been met, and the work continues.

1. Work on the first goal, specifying the framework, is essentially finished (see Section 1.2.2).
2. Very substantial progress has been made on creating the grammar of English; although the existing grammar is apparently adequate for some text generation tasks, some additions are planned (see Section 1.2.3.)
3. Progress on the third goal, although gratifying, is seriously incomplete. We have a notation and a design method for relating the grammar to prevailing text needs, and there are worked out examples which illustrate the methods (see Chapter 2), but the work so far is far from complete (see Section 1.2.4.)

1.2 A GRAMMAR FOR TEXT GENERATION--THE DESIGN

1.2.1 Overview of Nigel's Design

The creation of the Nigel program has not required radical revisions in systemic notation nor the reorganization of existing fragments of the grammar of English. The changes have been evolutionary and largely in the direction of making well-precedented ideas more explicit or detailed. The result has unified and extended significant amounts of existing work.

The level of explicitness is set primarily by the desire to incorporate the resulting methods and definitions in autonomous computer programs. Such programs cannot resort to human judgment, to intuitions about favorable cases, to well-known examples or incompletely specified principles. At each point in the generation process, the method must always specify an appropriate action to take next.

1.2.2 Programming the Systemic Framework

Systemic notation deals principally with three kinds of entities: 1) systems, 2) realizations of systemic choices (including function structures), and 3) lexical items. These three account for most of the notational devices, and the Nigel program has separate parts for each.

Comparing the systemic functional approach to a structural approach such as context-free grammar, ATNs or transformational grammar, the differences in style (and their effects on the programmed result) are profound. Although it is not possible to compare the approaches in depth here, we note several differences of interest to people more familiar with structural approaches:

1. Systems, which are most like structural rules, do not specify the order of constituents. Instead they are used to specify sets of features to be possessed by the grammatical construction as a whole.
2. The grammar typically pursues several independent lines of reasoning (or specification) whose results are then combined. This is particularly difficult to do in a structurally oriented grammar, which expresses the state of development in terms of categories of constituents.
3. In the systemic framework, all variability of the result, and hence all control, is in one kind of construct, the system. In other frameworks there is often variability from several sources: optional rules, disjunctive options within rules, optional constituents, order of application and so forth. For generation these would have to be coordinated by methods which lie outside of the grammar, but in the systemic grammar the coordination problem does not exist.

Nigel contains its entire grammar of English as a kind of data inside the program, not "procedurally embedded" as some previous grammars were [Winograd 72]. It can generate in any of three basic modes of choosing: random choice, manual choice and generation according to a programmed semantics (the Choosers, described in Section 1.2.4). These modes, and the finer controls on each, are part of a flexible set of provisions for testing the generator.[4]

1.2.2.1 Systems and gates

Each system has an ***input expression***, which encodes the entry conditions for the system.[5] During the generation, the program keeps track of the ***selection expression***, the set of features which have been chosen up to that point. Based on the selection expression, the program invokes the realization operations which are associated with each feature chosen. Nigel uses immediate realization rather than delayed realization, invoking each realization as soon as the associated feature is chosen. Systemic notation seems to work equally well whether realizations are performed immediately or in a collection after the grammar has been traversed. However, when the semantic operations are added, as described in Section 1.2.4, it becomes important that realization be immediate.

---

[4] In addition to the parts of the program devoted to generation, there are extensive parts devoted to printing various kinds of entities, representing them graphically, tracing the course of activity during generation, answering various questions about the program's content, and testing the integrity of the definitions as a collection. These together comprise a larger quantity of program than the parts which actually do the generation, but they are not described in this paper. Nigel is programmed in Interlisp, a dialect of Lisp, so it can run on any computer and operating system which can run Interlisp, including TOPS-20 and VAX systems.

[5] Input expressions are composed entirely of feature names, together with *And*, *Or* and parentheses. See the figures in Chapter 2 for examples.

In addition to the systems there are ***Gates.*** A gate can be thought of as an input expression which activates a particular grammatical feature, without choice.[6] These grammatical features are used just as those chosen in systems. Gates are most often used to perform realization in response to a collection of features.[7]

1.2.2.2 Realization operators

There are three groups of realization operators: those that build structure (in terms of grammatical functions), those that constrain order, and those that associate features with grammatical functions.

1. The realization operators which build structure are ***Insert, Conflate,*** and ***Expand.*** By repeated use of the structure building functions, the grammar is able to construct sets of ***function bundles***, also called ***fundles.***
2. Realization operators which constrain order are ***Partition, Order, OrderAtFront*** and ***OrderAtEnd.*** Partition constrains one function (hence one fundle) to be realized to the left of another, but does not constrain them to be adjacent. Order constrains just as Partition does, and in addition constrains the two to be realized adjacently. OrderAtFront constrains a function to be realized as the leftmost among the daughters of its mother, and OrderAtEnd symmetrically as rightmost.
3. Some operators associate features with functions. They are ***Preselect***, which associates a grammatical feature with a function (and hence with its fundle); ***Classify***, which associates a ***lexical feature*** with a function; ***OutClassify***, which associates a lexical feature with a function in a preventive way; and ***Lexify***, which forces a particular lexical item to be used to realize a function. Of these, OutClassify and Lexify are new, taking up roles previously filled by Classify. OutClassify restricts the realization of a function (and hence fundle) to be a lexical item which does not bear the named feature. This is useful for controlling items in exception categories (e.g., reflexives) in a localized, manageable way. Lexify allows the grammar to force selection of a particular item without having a special lexical feature for that purpose.

In addition to these realization operators, there is a set of ***Default Function Order Lists.*** These are lists of functions which will be ordered in particular ways by Nigel, provided that the functions on the lists occur in the structure, and that the realization operators have not already ordered those functions. A large proportion of the constraint of order is performed through the use of these lists.

The realization operations of the systemic framework, especially those having to do with order, have not been specified so explicitly before.

---

[6] See Figure 2-8 in Chapter 2 for examples and further discussion of the roles of gates.

[7] Each realization operation is associated with just one feature; there are no realization operations which depend on more than one feature, and no rules corresponding to Hudson's function realization rules. The gates facilitate eliminating this category of rules, with a net effect that the notation is more homogeneous.

1.2.2.3 The lexicon

The lexicon is defined as a set of arbitrary symbols, called ***word names***, such as "builten", associated with symbols called ***spellings***, the lexical items as they appear in text. In order to keep Nigel simple during its early development, there is no formal provision for morphology or for relations between items which arise from the same root.

Each word name has an associated set of ***lexical features.*** For convenience in managing large numbers of lexical features, there is a tree of lexical categories which can be manipulated by the processes used to enter definitions of lexical items into the program; this tree has no status in the theory.

Lexify selects items by word name; Classify and OutClassify operate on sets of items in terms of the lexical features.

Note that Nigel is not designed according to the view that the lexicon is homogeneous with the grammar, found at the most delicate positions. The semantic parts described below have special provisions for manipulating the lexicon.

1.2.3 The Grammar and Lexicon of English

Nigel's grammar is partly based on published sources, and is partly new. It has all been expressed in a single homogeneous notation, with consistent naming conventions and much care to avoid reusing names where identity is not intended. The grammar is organized as a single network, whose one entry point is used for generating all kinds of units. As of the summer of 1982, it contains about 220 systems, assigned informally to 28 collections. The relative emphasis on particular phenomena in the grammar can be judged loosely from the populations of the collections, below.

1. ADVERBIAL	(1 system)
2. ATTITUDE	(1 system)
3. CIRCUMSTANTIAL	(14 systems)
4. CLASSIFICATION	(3 systems)
5. CLAUSE-COMPLEX	(15 systems)
6. CONJUNCTION	(3 systems)
7. COUNT/NUMBER	(2 systems)
8. CULMINATION	(3 systems)
9. DEPENDENCY	(18 systems)
10. DETERMINATION	(23 systems)
11. ELLIPSIS	(1 system)
12. EPITHET	(4 systems)
13. MOOD	(15 systems)
14. NOUNTYPE	(2 systems)
15. POLARITY	(6 systems)
16. PRONOUN	(7 systems)
17. PP-SPATIOTEMPORAL	(8 systems)
18. PP-OTHER	(6 systems)
19. QUANTIFICATION	(5 systems)
20. QUALIFICATION	(6 systems)
21. RANKING	(4 systems)

22. TAG	(2 systems)
23. TENSE	(10 systems)
24. THEME	(11 systems)
25. NON-RELATIONAL TRANSITIVITY	(19 systems)
26. RELATIONAL TRANSITIVITY	(24 systems)
27. VERBALGROUP	(2 systems)
28. VOICE	(5 systems)

In developing the program, we have made linguistically significant progress in several areas: realization of order has turned out to be a rather complex matter, in which the programming has forced us to add substantially to existing precedents; we have identified a large number of integrity conditions which are applicable to systemic grammars, and programmed them as tests; the grammar has been worked on extensively in the area of tense.

Nigel's lexicon is designed for test purposes rather than coverage of any particular generation task. It currently recognizes 130 lexical features, and it has about 2000 lexical items in about 580 distinct categories (combinations of features).

1.2.4 Choosers--The Grammar's Semantics

The most novel part of Nigel from a systemicist's point of view is the semantics of the grammar. The goal identified above was to "specify how the grammar can be regulated effectively by the prevailing text need." Just as the grammar and the resulting text are both very complex, so is the text need. In fact, these kinds of complexity actually reflect the complexity of the text need which gave rise to the text. The grammar must respond selectively to those elements of the need which are represented by the unit being generated at the moment.

Except for lexical choice, all variability in Nigel's generated result comes from variability of choice in the grammar. Generating an appropriate structure consists entirely of making the choices in each system appropriately; there is no other locus of variation. The semantics of the grammar must therefore be a semantics of the individual systems. Choices must be made in each system according to the appropriate elements of the prevailing need.

In Nigel this semantic control is localized to the systems themselves. For each system, a procedure is defined which can declare the appropriate choice in the system. When the system is entered, the procedure is followed to discover the appropriate choice. Such a procedure is called a ***chooser*** (or "choice expert"). The chooser is the semantic account of the system, the description of the circumstances under which each choice is appropriate.

To specify the semantics of the choices, we needed a notation for the choosers as procedures. This paper describes that notation briefly and informally. Its use is exemplified in the Nigel demonstration in Chapter 2 and developed in more detail in another report [Mann 82].

To gain access to the details of the need, the choosers must in some sense ask questions about particular entities. For example, to decide between Singular and Plural in creating a NominalGroup, the Number chooser (the chooser for the Number system) must be able to ask whether a particular entity (already identified elsewhere as the entity the NominalGroup represents) is unitary or multiple. That knowledge resides outside of Nigel, in the ***environment***.

The environment is regarded informally as being composed of three disjoint regions:

1. The ***Knowledge Base***, consisting of information which existed prior to the text need;
2. The ***Text Plan***, consisting of information which was created in response to the text need, but before the grammar was entered;
3. The ***Text Services***, consisting of information which is available on demand, without anticipation.

The chooser must have access to a stock of symbols representing entities in the environment. Such symbols are called ***hubs***. In the course of generation, hubs are associated with grammatical functions. These associations are kept in a ***Function Association Table***. This table is used to reaccess information in the environment. For example, in choosing pronouns the choosers will ask questions about the multiplicity of an entity which is associated with the THING function in the Function Association Table. Later they may ask about the gender of the same entity, again accessing it through its associaiton with THING. This use of grammatical functions is an extension of previous uses. It has several fortunate consequences: relations between referring phrases and the concepts being referred to are captured in the Function Association Table. For example, the function representing the NominalGroup as a whole is associated with the hub which represents the thing being referred to in the environment. Similarly for possessive determiners, the grammatical function for the determiner is associated with the hub for the possessor. In cases of ellipsis, the function which would have expressed the ellipsed element (if it had been inserted into the structure) carries the appropriate hub whether ellipsis takes place or not, so that there is a formal identification of the entity being ellipsed.

It is convenient to define choosers in such a way that they have the form of a tree. For any particular case, a single path of operations is traversed.

Choosers are defined principally in terms of the following operations:

1. ***Ask*** presents an inquiry to the environment. The inquiry has a fixed predetermined set of possible responses, each corresponding to a branch of the path in the chooser.
2. ***Identify*** presents an inquiry to the environment. The set of responses is open-ended. The response is put in the Function Association Table, associated with a grammatical function which is given (in addition to the inquiry) as a parameter to the Identify operator.[8]
3. ***Choose*** declares a choice.
4. ***CopyHub*** transfers an association of a hub from one grammatical function to another.[9]

An ***inquiry*** consists of an ***inquiry operator*** and a sequence of ***inquiry parameters***. Each inquiry parameter is a grammatical function, and it represents (via the Function Association Table) the entities in the environment which the grammar is inquiring about. The operators are defined in such a way that they have both formal and informal modes of expression. Informally, each inquiry is a

---

[8] See Chapter 2 for an explanation and example of its use.

[9] There are three others which have some linguistic significance: Pledge, TermPledge, and ChoiceError. These are necessary but do not play a central role. They are named here just to indicate that the chooser notation is very simple.

predefined question, in English, which represents the issue that the inquiry is intended to resolve for any chooser that uses it. Formally, the inquiry shows how systemic choices depend on facts about particular grammatical functions, and in particular restricts the account of a particular choice to be responsive to a well-constrained, well-identified collection of facts. Both the informal English form of the inquiry and the corresponding formal expression are regarded as parts of the semantic theory expressed by the choosers which use the inquiry.

The entire collection of inquiries for a grammar is a definition of the semantic scope to which the grammar is responsive at its level of delicacy.

Figure 1-1 shows the chooser for the ProcessType system, whose grammatical feature alternatives are Relational, Mental, Verbal and Material.

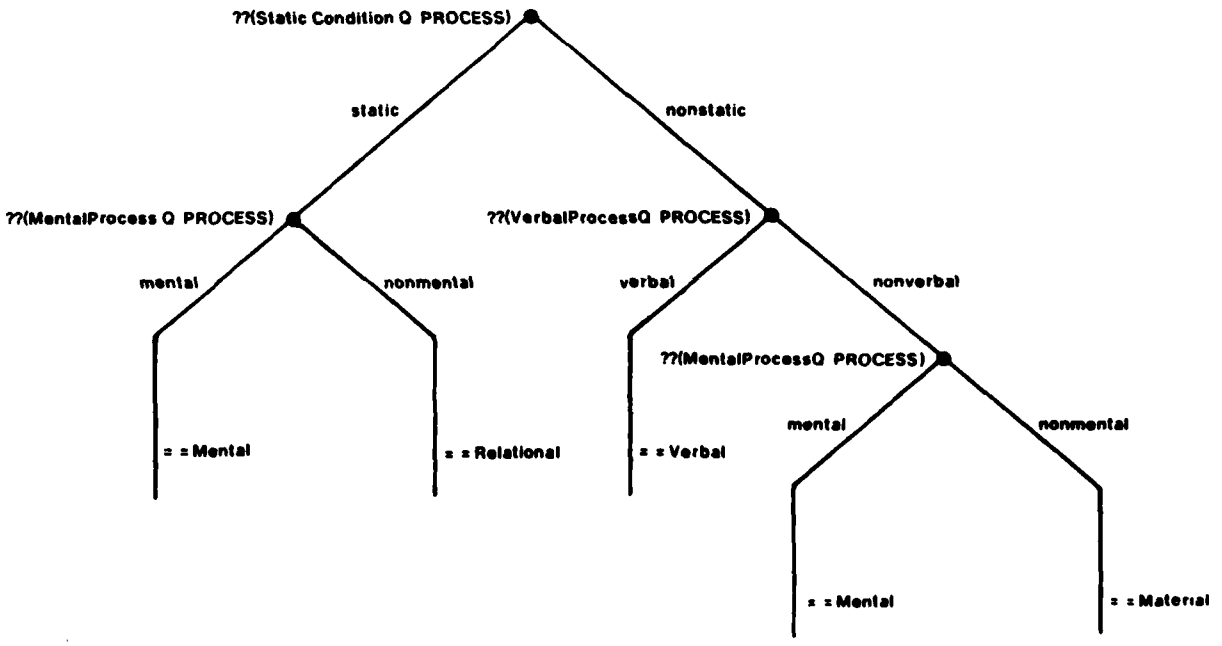


Figure 1-1: The chooser of the ProcessType system

Notice that in the ProcessType chooser, although there are only four possible choices, there are five paths through the chooser, because Mental processes can be identified in two different ways: those which represent states of affairs and those which do not. The number of termination points of a chooser often exceeds the number of choices available.

Figure 1-2 shows the graphic representational elements which are used to represent choosers.

Table 1-1 shows the English forms of the questions being asked in the ProcessType chooser.

Table 1-1: English forms of the inquiry operators for the ProcessType chooser

StaticConditionQ	**Does the process PROCESS represent a static condition or state of being?**
VerbalProcessQ	**Does the process PROCESS represent symbolic communication of a kind which could have an addressee?**
MentalProcessQ	**Is PROCESS a process of comprehension, recognition, belief, perception, deduction, remembering, evaluation or mental reaction?**

??	**Ask**
<?	**Identify**
= =	**Choose**
<<	**CopyHub**
!#	**Pledge**
!*	**TermPledge**

Figure 1-2: Graphic representation elements for choosers

The sequence of inquiries which the choosers present to the environment, together with its responses, creates a dialogue. The unit generated can be seen as being formed out of a negotiation between the choosers and the environment. This is a particularly instructive way to view the grammar and its semantics, since it identifies clearly what assumptions are being made and what dependencies there are between the unit and the environment's representation of the text need. (This is the kind of dialogue represented in Chapter 2.)

It also leads to a particularly helpful mode of testing the choosers, which proceeds as follows:

- From some natural source of text, select a phrase or clause which appears to be within the syntactic capability of the grammar.
- Attempt to generate it using Nigel, in a mode in which the answers to the inquiries come from the user of the program. Answer all of the inquiries straightforwardly according to the selected unit and the context in which it was found.
- See whether the generated result is identical to the original.

This method, which has been called the "method of honest muddling," usually reveals some discrepancy between the way the choosers were intended to work and the way the inquiries are defined. This leads in turn to improved definitions for choosers and their inquiries, and often to a local reconceptualization as well.

The grammar performs the final steps in the generation process. It must complete the surface form of the text, but there is a great deal of preparation necessary before it is appropriate for the grammar to start its work. Penman's design calls for many kinds of activities under the umbrella of "text planning" to provide the necessary support. Work on Nigel is proceeding in parallel with other work intended to create text planning processes.

1.3 USES FOR NIGEL

The activity of defining Nigel, especially its semantic parts, is productive in its own right. It creates interesting descriptions and proposals about the nature of English and the meaning of syntactic alternatives, as well as new notational devices. But given Nigel as a program, containing a full complement of choosers, inquiry operators and related entities, new possibilities for linguistic investigation also arise.[10]

One can run the program to see what it generates. Apart from such a test, there seems to be no practical way to find out whether the grammar produces unintended combinations of functions, structures or uses of lexical items.

One can also see how it fails to generate. In the early phases of giving Nigel a grammar of English, many contradictions were found within the grammar. Ordering was sometimes so strong that it produced cycles, so that a constituent was told to precede itself. Incompatible preselections were attached to fundles, requiring that more than one of the options be taken in a subsequently entered system, and so forth. It appears that there is a natural tendency to write the grammar with excessive homogeneity, not allowing for possible exception cases.

One can attempt to replicate text, identifying the semantic conditions which would necessarily have to be present to lead to the particular text at hand. This gives an objective way to assign some rather subtle meanings to text.

---

[10] It is our intention eventually to make Nigel available for teaching, research, development and computational application.

On another scale, the whole project can be regarded as a single experiment, a test of the functionalism of the systemic framework, and of its identification of the functions of English.

Nigel accounts for certain kinds of functions and relies on as yet undefined text planning processes for others. This decomposition is significant as a claim about kinds of functions and how they group into logically dependent sequences. The demands for text plan elements challenge future research to create processes which can actually plan texts in a compatible way.

Nigel's grammar, separated from program, choosers and lexicon, is usable in all the traditional ways. Because it is in a consistent notation and has been tested extensively, it has some advantages for educational and linguistic research uses.

In artificial intelligence, there is a need for priorities and guidance in the design of new knowledge representation notations. The inquiry operators of Nigel are a particularly interesting proposal as a set of distinctions already embodied in a mature, evolved knowledge notation, English, and encodable in other knowledge notations as well. For example, the inquiry operators suggest that a notation for knowledge should be able to represent objects and actions, and should be able to distinguish between definite existence, hypothetical existence, conjectural existence and nonexistence of actions. These are presently rather high expectations for artificial intelligence knowledge representations.

The choosers seem particularly interesting as an expository device in teaching English to nonnative speakers. The procedures document the syntactic devices of the language in a way which avoids the problems of induction from a collection of examples.

There are other practical possibilities. Computer programs with access to complex collections of information may be enabled to express that information in one or more human languages to meet practical human needs for the information.

1.4 SUMMARY

As part of an effort to define a text generation process, a programmed systemic grammar called Nigel has been created. Systemic notation, a grammar of English, a semantic notation, and a semantics for English are all included as distinct parts of Nigel. When Nigel has been completed it will be useful for a variety of research, educational, and practical purposes.

REFERENCES

[Berry 75] Berry, M., *Introduction to Systemic Linguistics: Structures and Systems*, Batsford, London, 1975.

[Berry 77] Berry, M., *Introduction to Systemic Linguistics: Levels and Links*, Batsford, London, 1977.

[Davey 79] Davey, A., *Discourse Production*, Edinburgh University Press, Edinburgh, 1979.

[de Joia 80] de Joia, A., and A. Stenton, *Terms in Systemic Linguistics*, Batsford Academic and Educational, Ltd., London, 1980.

[Fawcett 80] Fawcett, R. P., *Exeter Linguistic Studies.* Volume 3: *Cognitive Linguistics and Social Interaction,* Julius Groos Verlag Heidelberg and Exeter University, 1980.

[Halliday 76a] Halliday, M. A. K., and R. Hasan, *Cohesion in English,* Longman, London, 1976. English Language Series, Title No. 9.

[Halliday 76b] Halliday, M. A. K., *System and Function in Language,* Oxford University Press, London, 1976.

[Halliday 81] Halliday, M.A.K., and J. R. Tastin (eds.), *Readings in Systemic Linguistics,* Batsford, London, 1981.

[Hudson 76] Hudson, R. A., *Arguments for a Non-Transformational Grammar,* University of Chicago Press, Chicago, 1976.

[Mann 81] Mann, W. C., et al., *Text Generation: The State of the Art and the Literature*, USC/Information Sciences Institute, RR-81-101, December 1981. To appear in *American Journal of Computational Linguistics.*

[Mann 82] Mann, W. C., *The Anatomy of a Systemic Choice*, USC/Information Sciences Institute, RR-82-104, October 1982.

[McDonald 80] McDonald, D. D., *Natural Language Production as a Process of Decision-Making Under Constraints*, Ph.D. thesis, Massachusetts Institute of Technology, 1980.

[McKeown 82] McKeown, K. R., *Generating Natural Language Text in Response to Questions about Database Structure*, Ph.D. thesis, University of Pennsylvania, 1982.

[Winograd 72] Winograd, T., *Understanding Natural Language,* Academic Press, Edinburgh, 1972.

2. A DEMONSTRATION OF THE NIGEL TEXT GENERATION COMPUTER PROGRAM

William C. Mann
Christian M. I. M. Matthiessen

ABSTRACT

How can a particular sentence be generated given the intention to communicate? This paper demonstrates a systemic grammar in use and its associated semantics, which is the core of a text generation program. The demonstration is a dramatization in which the actors play the roles of the major components of the program. The generation drama consists of the interactions of these components in the generation of the sentence *This gazebo was built by Sir Christopher Wren.* By the end of the generation, almost all major aspects of the grammar and its semantics will have been illustrated and explained in context.

The paper is a transcript of the actual demonstration; the mode is "written to be spoken to an audience."[11]

2.1 INTRODUCTION TO THE DEMONSTRATION

Although the title of this session announces a computer program demonstration, we have no computer here, and you won't have a chance to see a computer producing its outputs. That may be a relief, since computer program demonstrations can easily become obscure and mysterious.

We do have a computer program called Nigel that contains a large systemic grammar of English and is capable of generating a wide range of sentences. But instead of just running it for you, we'd like to give a more comprehensible view of Nigel, by another kind of representation of what it does--a generation drama.

Nigel was built as part of a larger text generation system called Penman. The research goal for Penman is to model the generation of fluent multiparagraph texts, in English, which correspond to given intentions to communicate. This is obviously an ambitious goal, since it involves in at least a token way many of the functions of authorship. The role of the computer is, in most ways, inessential. It gives us some methodological biases and serves as a tool for various kinds of experiments. But what you will see is not computational in any essential way.

The only part of Penman which has been extensively developed so far is its grammar.

Two kinds of activities have taken most of our time in the last two years or so since we started building the grammar.

1. specifying all of the details well enough, and

---

[11] The original audience was a conference on systemic linguistics.

2. making the grammar responsive to the intent, purpose or achievement goal of the speaker.

This morning we're going to concentrate on the second of these, i.e., making the grammar responsive to intent. Since the systemic framework is functionally organized, this is a problem of extending that functionalism so that it is very explicit. Consequently, we are really serving a much broader collection of goals than the origins of the project would suggest. This task has not turned out at all to be just an application of computation to the systemic framework. In some ways it has tested the functionalism of the systemic framework as a whole.

Our goal in the demonstration is to convey an informal understanding of a method for defining the semantics of the choices in a systemic grammar. We are not trying to convey any formal scheme. Rather, we would like to say: Here is an approach to defining the semantics of a systemic grammar, one which leads to examinable results.

We will proceed by first sketching the approach, and then showing how a particular sentence can be generated.

2.2 AN APPROACH TO THE SEMANTICS OF A SYSTEMIC GRAMMAR

How can a text be produced from an intention to communicate? Our general view is that there are text planning processes which are directly responsive to speakers' intentions to communicate, and that for monologue it is helpful to view them as logically prior to the activity of the grammar.

Presupposing this, how can a text be produced from a planful response to an intention to communicate? This is a question of method. The question can be answered by specifying a method of generation. Since methods are inherently processlike, and since, from our knowledge of language, we expect that the method will have a complex abstract structure, it must be specified, not by one process, but by a collection of processes which can work together to create the text. (Here we are using "process" in the sense of procedure, not in its grammatical sense, nor in the sense Hjelmslev used it.)

The definitional framework which is the basis of the Nigel grammar is just such a collection of processes. It is systemic and functional, and it has provisions for realization. It is based on the work of Halliday and other systemicists, with additional work by Halliday and major contributions by Christian Matthiessen and Bill Mann. The chooser and inquiry framework which we will describe was developed by Bill Mann and Christian Matthiessen.

Because the systemic framework is used, all grammatical control is in the choices; the focal concern is getting the choices right. This means more than just getting the range of possibilities right. The potential of the language must be applied, in each particular case, in a way which conforms to the speaker's intent.

To get the choices right we have added another stratum, a semantic one which adds two principal kinds of devices to the basic grammatical framework:

1. A set of choosers (also called "choice experts")

2. A set of inquiries

Choosers are explicit processes that are able to examine the case at hand and make a systemic choice. There is a separate, independent chooser for each system. So, the system expresses a particular set of alternatives of linguistic potential, and the chooser expresses the circumstances under which each particular choice is made. A chooser acts by performing a sequence of steps that is specified by its definition as a process.

The second kind of device, the inquiries, is questions that choosers can ask to help them understand the present case. Around Nigel there is a boundary. A chooser will typically present two or three inquiries to the region beyond this boundary, what we call the Environment.[12] This environment has three parts:

1. **Knowledge base:** knowledge which existed before the intention to communicate.
2. **Text plan:** knowledge which was developed in response to the intention to communicate, but before entering the grammar.
3. **Text services:** knowledge which is available to the grammar on demand.

The grammar has no direct access to any of these kinds of knowledge, because they are outside of the boundary. However, the boundary is really an interface, so the grammar can present inquiries at the interface, and receive answers. This is how it finds out everything it needs to know about the present case. We have tried to define all of these inquiries so that they do not demand any grammatical knowledge from the environment.

There are two forms of each inquiry: an English question and a corresponding formal inquiry form.

Because inquiries have an English form, the actions at the interface can be understood as a dialogue between the grammar, represented by the choosers of all the systems, and the environment.

2.3 PRELIMINARIES TO THE GENERATION PROCESS

The Nigel computer program does all of the kinds of things that go on inside the boundary. It enters systems, runs chooser processes, presents inquiries to the interface, receives answers, makes choices and performs realizations.

As our demonstration we are going to present one of Nigel's dialogues, derived from a computer run in which a sentence was generated. We won't be using the computer directly; instead we will have people representing the various parts.

In Nigel, and so in our demonstration, initiative comes from the grammar. The general control on what happens comes from the entry conditions of the systems. It is not the case that the semantic

---

[12] There is an obvious correspondence between "environment" and the Malinowskian/Firthian notion of context. However, the term "environment" will be used here, since it does not imply a detailed Firthian theory of context.

stratum has its own control, does its work and presents the results to the grammar for realization. Instead, it is controlled by the entry conditions of the systems. This means that grammar and semantics work in synchrony.

To keep the various activities distinct, we have assigned different kinds of activities to be played as roles by different people:

- **Environment:** The environment is represented by Ruqaiya Hasan. She has three books which she consults in order to answer inquiries. They are called Knowledge Base, Text Plan and Text Services, representing the three kinds of knowledge in the environment.
- **Choosers:** There is one individual, Bill Mann, representing all of the different independent choosers: he asks questions, keeps track of some answers in the Function Association Table, and chooses.
- **Grammar:** All of the systems of the grammar are represented by Michael Halliday. He enters systems and keeps track of the selection expression.
- **Realizer:** All of the realization operations are also represented by Michael Halliday. He shows each realization as soon as it becomes definite.

Each of these represents very closely what the computer program does.

Grammar and Realizer represent the grammatical component or stratum. Grammar interacts with the Choosers, which represent the semantics collectively. The Choosers also interact with Environment. There is no direct interaction between Grammar and Environment: the Choosers mediate as an *inter-level*. The possible interactions are summarized diagrammatically in Figure 2-1.

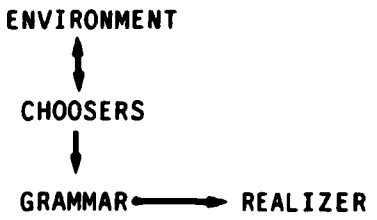


Figure 2-1: Role interactions

There is an additional character in our drama, the narrator (played by Christian Matthiessen), whose function is to give comments at a meta-level.

Some activity will be suppressed to avoid tedious repetition. The grammar provides for many adjuncts, and for many modifiers in NominalGroups, but only a few will be shown. We will not try to justify either content or methodology.

The work has involved a fair amount of innovation, and so, in presenting it to you, there is a tradeoff between how much we can show of *how* we do things, on the one hand, and how much we can explain *why* we do them that particular way, on the other hand. The more we justify, the less we can show of our methods. This particular session will have a minimum of justification and, we hope, a maximum of useful demonstration of methods.

The situation as we enter the grammar is that the text plan already exists. This plan is indexed into the knowledge of the subject matter, and both are available in the environment. The sentence we are generating is the second sentence in the paragraph represented in Figure 2-2 (derived from [Halliday 70]).

In Greenwich, in South East London, there is a small brick gazebo. **This gazebo was built by Sir Christopher Wren.** It is a rather undistinguished structure, which might have been a task set for homework when he was at school.

Figure 2-2: The gazebo text

The order of entering the systems which we will meet is exactly the order that the computer program actually employed. Often there were several systems which could have been entered, and the particular selection was arbitrary. This leads to a somewhat disconnected style, in which, for example, the program gets part way through developing tense, hops off to do other things, and then returns to complete the tense development much later. For purposes of presentation, the generation process will be divided into thirteen sections, each represented by a section of the system network of Nigel. Each section comprises one or more systems. The sections are as follows:

1. RANK: Rank; CLASS: ClauseClass; COMPLEXITY: ClauseComplexity
2. SECONDARY TENSE: NonDeicticPresent, ZNonDeicticTense
3. TRANSITIVITY I: Location, Manner; Agency, ProcessType, AgentInsert
4. TRANSITIVITY II (MATERIAL TRANSITIVITY): DoingType, GoalInsertConflate
5. CLAUSE VOICE: EffectiveVoice, Agentivity
6. VERBAL VOICE: LexverbPasspart
7. MOOD I: Dependence, MoodType
8. MOOD II: IndicativeType, DeclarativeTag
9. THEMATIZATION: ThemeMarkingDeclarative
10. DEICTICITY: Deicticity, PrimaryTense
11. POLARITY: Polarity, PolarityMarkingPositive
12. SUBJECT PERSON: ThingSubject, IndicativeOtherSubjectNumber
13. LEXICALVERB TERMRESOLUTION: LexicalverbTermResolution

As features are chosen in these systems, the Grammar will accumulate a selection expression. For each section, we will show the current selection expression as it has been developed up to that point in the generation process. After the completion of the generation process, its results will be summarized and a complete selection expression is given below in Section 2.4.13, (Figure 2-27).

As we go along generating *This gazebo was built by Sir Christopher Wren*, the Realizer will build the clause structure step by step. The final result, the complete clause structure, is also presented in Section 2.6 (in Figure 2-28) and you can trace each step the Realizer takes in the figure.

Before we start generating, we need a couple of additional terms. There are elements of the text plan in the environment, and also elements of prior knowledge. The environment is always able to assign an arbitrary name to any such element, and to give that name as an answer to an inquiry. These names are called *Hubs*. Arbitrary mnemonic names are used as hubs, but the hubs should not be thought of as lexical items.

The grammar's set of symbols does not include any hubs, so it does not know any of the specific symbols which represent knowledge in the environment. It does not even know the symbol which represents the identity of the speaker. It keeps track of the environment's symbols by associating them with symbols of its own, starting over each time the grammar is entered. The place where it keeps these associations is called the Function Association Table. The Function Association Table will be developed throughout the generation process (Tables 2-1, 2-2, 2-3, and so on). In these tables, the right column lists grammatical functions and the left column lists a hub for each grammatical function, lined up according to its association. For example, the first association is (hub:) WREN-GAZEBO (associated with function:) ONUS.

Whenever the grammar is entered, there is one symbol, ONUS, which already has an association in the Function Association Table (as in the present example in Table 2-1 below). This grammatical function always represents the largest unit to be generated on the present entry to the grammar. In the demonstration case, the hub associated with ONUS will be the environment's name for the plan to generate the second sentence of the paragraph.

2.4 THE GENERATION PROCESS

NARRATOR: ***The drama begins with an association of WREN-GAZEBO with ONUS in the Function Association Table, which is presented in Table 2-1. The grammar can start***

Table 2-1: Function Association Table: initial state

HUBS	GRAMMATICAL FUNCTIONS
WREN-GAZEBO	ONUS

2.4.1 Rank, Class, and Complexity

2.4.1.1 Rank

GRAMMAR: We now enter the Rank system (see Figure 2-3). The choice options are Clauses, GroupsPhrases, and Words.

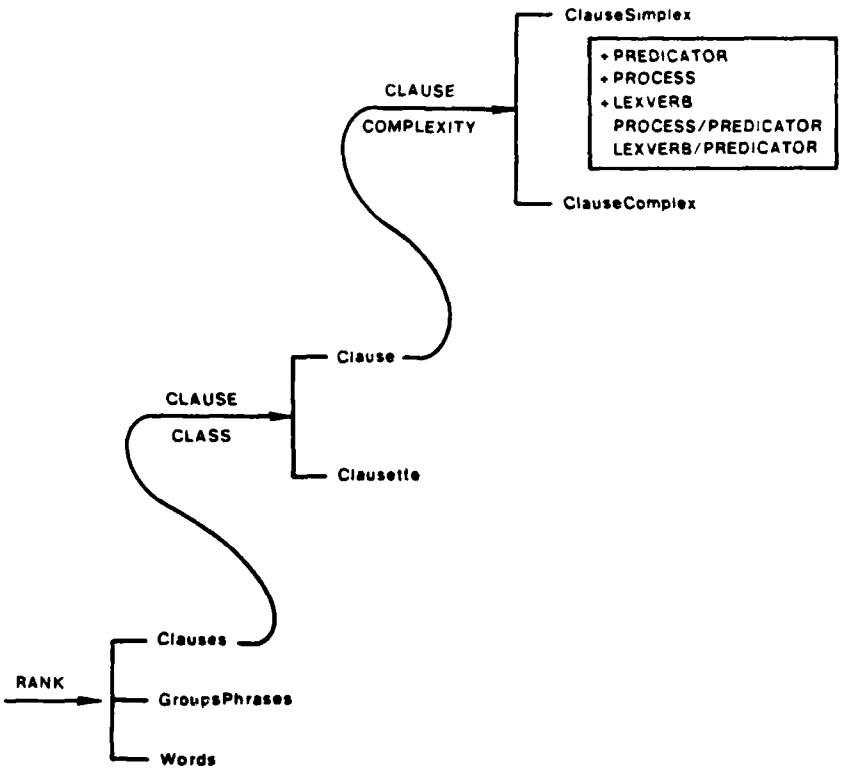


Figure 2-3: Rank, Class, and Complexity

NARRATOR: ***The grammar is a single network grammar, so rank is the least delicate system in this network. Consequently, the Rank system is always the first system entered whenever the grammar is entered, no matter what grammatical unit is being generated.***

CHOOSER: Does WREN-GAZEBO (ONUS)[13] have an illocutionary force?

ENVIRONMENT: Yes, it has an illocutionary force.

CHOOSER: Then I choose feature Clauses, which is the first addition to the selection expression in Figure 2-4.

```
Clauses, Clause, ClauseSimplex
```

Figure 2-4: Selection Expression, stage: (1)

NARRATOR: ***The rest of the questions by this chooser are for the purpose of adding information to the association table. Other choosers, for example the ClauseClass chooser right below, can use this information.***

---

[13]The notation WREN-GAZEBO (ONUS) means "the hub WREN-GAZEBO associated with the function ONUS"

CHOOSER: What is the concept which is the speech act aspect of WREN-GAZEBO (ONUS)?

ENVIRONMENT: The speech act aspect is called WREN-GAZEBO-STATEMENT.

CHOOSER: I'll put the association of WREN-GAZEBO-STATEMENT with SPEECH-ACT in Table 2-2.

Table 2-2: Function Association Table, stage: (1)

HUBS	GRAMMATICAL FUNCTIONS
WREN-GAZEBO-STATEMENT	SPEECH-ACT
NOW	SPEAKING-TIME
GAZEBO-BUILDING	PROCESS
HISTORIC-TIME	EVENT-TIME

CHOOSER: ... and I'm finished in this chooser.

NARRATOR: ***The preceding conversation illustrates much of the mechanism: The grammar controls the entry of systems. Given that a system is entered, the chooser of that system will ask one or more questions of the environment and choose according to the answers. There may be further questions to establish associations between a hub and a grammatical function. When the chooser of a system has been exhausted, it announces that it has finished.***

2.4.1.2 ClauseClass

GRAMMAR: We now enter the **ClauseClass** system. The choice options are Clause and Clausette.

NARRATOR: ***The terms Clause and Clausette correspond to the terms major clause and minor clause.***

CHOOSER: Is the speech act WREN-GAZEBO-STATEMENT (SPEECH-ACT) one which has a propositional parameter?

ENVIRONMENT: Yes, the act has a propositional parameter.

CHOOSER: Then I choose feature Clause (added to the selection expression in Figure 2-4).

NARRATOR: ***Again, the question which follows gathers information for later use by systems developing tense; see Section 2.4.2 below.***

CHOOSER: What concept represents the current time, the time at which the language is generated?

ENVIRONMENT: It's called NOW.

CHOOSER: I'll put the association of NOW with SPEAKING-TIME in Table 2-2.

CHOOSER: ... and I'm finished in this chooser.

2.4.1.3 ClauseComplexity

GRAMMAR: We now enter the **ClauseComplexity** system. The choice options are ClauseSimplex and ClauseComplex.

CHOOSER: Does expression of WREN-GAZEBO (ONUS) represent a multiplicity of related processes rather than a single process?

ENVIRONMENT: It's a single process.

CHOOSER: Then I choose feature ClauseSimplex (added to the selection expression in Figure 2-4).

CHOOSER: What concept is the process aspect of WREN-GAZEBO (ONUS), i.e., the one which governs the main verb?

ENVIRONMENT: It's called GAZEBO-BUILDING.

CHOOSER: I'll put the association of GAZEBO-BUILDING with PROCESS in Table 2-2.

CHOOSER: What concept represents the time of occurrence or the restricted portion of the time of occurrence of GAZEBO-BUILDING (PROCESS) which this mention of it has in view?

NARRATOR: ***This is a somewhat obscure question. It is simply asking for a symbol to represent the time at which the gazebo was built. In general,*** inquiries ***are intended to encode the semantics of a particular system and are phrased so that they will be sufficiently explicit.***

ENVIRONMENT: All right, it occured at HISTORIC-TIME.

CHOOSER: I'll put the association of HISTORIC-TIME with EVENT-TIME in Table 2-2.

CHOOSER: Now, what is the set of applicable lexical items which are denotationally appropriate for expressing GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: The answer is a set of 15 terms (see Figure 2-5): the first 5 are "build," "built" (as a past form), "built" (as an *en*-participle), "building," and "builds." The other 10 are the corresponding forms of "create" and "construct."

build	create	construct
builds	creates	constructs
built	created	constructed
built	**created**	**constructed**
building	creating	constructing

Figure 2-5: The initial termset for PROCESS

NARRATOR: ***Nigel keeps track of sets of candidate lexical items for various grammatical functions. In this case, PROCESS will be associated with these 15 terms. Abstractly, there are two ways in which sets of candidate lexical items are constrained and denotational appropriateness is the first kind of constraint applied. Then grammatical constraints--such as the requirement that the lexical item be an en-participle--are used to filter the set of denotationally appropriate terms. Finally, the remaining indeterminacy is resolved on some other basis, such as register or frequency.***

In this case, the grammar will apply grammatical constraints to reduce this set of 15 terms to 3 en-participles, and the final selection will be made from that set of 3.

CHOOSER: ... and I'm finished in this chooser.

NARRATOR: ***We are about to see the realization process in operation. There are 9 types of realization represented in this example. (I) One group builds and specifies the function structure of the unit being generated. These fall into two subgroups. The first includes Insert, Conflate, and Expand. Insert adds a new function to the structure (e.g., (Insert SUBJECT)). Conflate operates on two functions and specifies that they are the same constituent (e.g., (Conflate SUBJECT AGENT)). Expand specifies a constituent relationship between two functions (e.g., (Expand MOOD SUBJECT)). The second subgroup is made up of ordering operators, Order, OrderAtEnd, and OrderAtFront.***

(II) The second group consists of operators that specify a grammatical feature, a lexical feature or a lexical item that realizes a functional

constituent. Preselect specifies a choice to be made in realizing a constituent in a subsequent pass through the grammar. Classify specifies a required lexical feature and Lexify specifies a lexical item to be used.

You can follow the steps in the realization process in the complete structure to be generated displayed in Figure 2-28.

The Realizer will now perform realization for the features already selected.

REALIZER: Insert PREDICATOR.[14]

REALIZER: Insert PROCESS.

REALIZER: Insert LEXVERB.

REALIZER: Conflate LEXVERB PROCESS.

REALIZER: Conflate PREDICATOR LEXVERB.

2.4.2 Secondary Tense

2.4.2.1 NonDeicticPresent

GRAMMAR: We now enter the **NonDeicticPresent** system--see Figure 2-7. The choice options are NoNonDeicticPresent and NonDeicticPresent.

---

[14] In this script, realizations are shown in their formal forms. The realizer person reads these relations as follows:

- Insert FF: We now insert the function FF in the structure.
- Conflate FF GG: We now conflate function FF with function GG. Symbol: +FF
- Expand NN TT: We now expand function NN to have as a constituent function TT. Symbol: NN(TT)
- Classify FF LL: We now classify function FF with lexical feature LL. Symbol: FF! LL
- Preselect FF PP: We now preselect function FF to use grammatical feature PP. Symbol: FF: PP
- Lexify FF II: We now commit to using lexical item II to express function FF. Symbol: FF = II
- Order FF GG: We now order function FF before and adjacent to function GG in the structure. Symbol: FF↑GG
- OrderAtEnd FF: We now order function FF to be the rightmost among those that descend from its mother. Symbol: FF ↑ L
- OrderAtFront FF: We now order function FF to be the leftmost among those that descend from its mother. Symbol: # ↑FF

NARRATOR: ***These are the feature names from the grammar. Others might call them Progressive and NonProgressive.***

CHOOSER: Is the activity represented by GAZEBO-BUILDING (PROCESS) complete at the time represented by HISTORIC-TIME (EVENT-TIME)?

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense
```

Figure 2-6: Selection Expression, stage: (2)

ENVIRONMENT: It is complete.

CHOOSER: Then I choose feature NoNonDeicticPresent and the feature is added to the selection expression (see Figure 2-6).

NARRATOR: ***The NonDeicticPresent chooser is an example of a chooser that is in the early stages of development; it is a placeholder for a more sophisticated chooser.***

CHOOSER: ... and I'm finished in this chooser.

2.4.2.2 ZNonDeicticTense

GRAMMAR: We now enter the **ZNonDeicticTense** system (a secondary tense). The choice options are PastZ and FutureZ.

NARRATOR: ***The tense we meet first in our development of the tense combination for the clause is ZTense; the Z indicates that its realization is rightmost in the tense expression.***

CHOOSER: What concept represents the time to which GAZEBO-BUILDING (PROCESS) is relevant?

NARRATOR: ***This chooser question is part of a treatment of tense that fleshes out the semantics implicit in Halliday's grammatical analysis. It also draws on Reichenbach, particularly for the notion of relevant time (which he calls reference time), and adds some innovations of our own.***

ENVIRONMENT: It's HISTORIC-TIME.

CHOOSER: I'll put the association of HISTORIC-TIME with RELEVANT-TIME in Table 2-3.

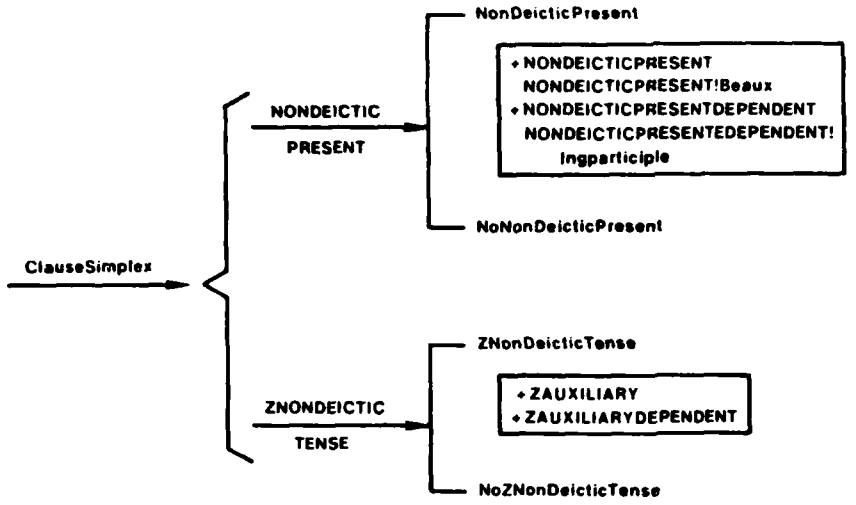


Figure 2-7: Secondary tense

Table 2-3: Function Association Table, stage: (2)

HUBS	GRAMMATICAL FUNCTIONS
WREN-GAZEBO	ONUS
WREN-GAZEBO-STATEMENT	SPEECH-ACT
NOW	SPEAKING-TIME
GAZEBO-BUILDING	PROCESS
HISTORIC-TIME	EVENT-TIME
HISTORIC-TIME	RELEVANT-TIME

CHOOSER: Is HISTORIC-TIME (RELEVANT-TIME) identical to HISTORIC-TIME (EVENT-TIME)?

ENVIRONMENT: Yes, they are the same.

CHOOSER: OK, I choose feature NoZNonDeicticTense (added to the selection expression in Figure 2-6).

NARRATOR: ***The grammar of tense starts with the temporal location of the process (event/state) and specifies it with respect to the time of speaking, one***

step at a time--i.e., one tense at a time. Complex tenses are built up just as Halliday's names for them are, for example, past-in-future-in-present. Here there is no secondary tense: The realization of this particular feature, NoZNonDeicticTense, is zero, and the realizer will keep quiet.

CHOOSER: ... and I'm finished in this chooser.

2.4.3 Transitivity I

2.4.3.1 Agency

GRAMMAR: We now enter the **Agency** system-- see Figure 2-8. The choice options are Middle and Effective.

CHOOSER: Is the process GAZEBO-BUILDING (PROCESS) one which conceptually has some sort of entity which causes the process to occur?

ENVIRONMENT: Yes, it is caused.

CHOOSER: What is the entity which caused or causes GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: The causer is SIR-CHRISTOPHER.

CHOOSER: I'll put the association of SIR-CHRISTOPHER with CAUSER in Table 2-4.

CHOOSER: Now, is it preferable to mention the causative relation between SIR-CHRISTOPHER (CAUSER) and the GAZEBO-BUILDING (PROCESS) process?

ENVIRONMENT: Mention it.

CHOOSER: All right, I choose feature Effective, which is added to the selection expression in Figure 2-9.

CHOOSER: ... and I'm finished in this chooser.

REALIZER: Classify PROCESS Effective

NARRATOR: ***Classify is one of the realization operators, comparable to Preselect. It means: Require this lexical feature on the item realizing that function.***

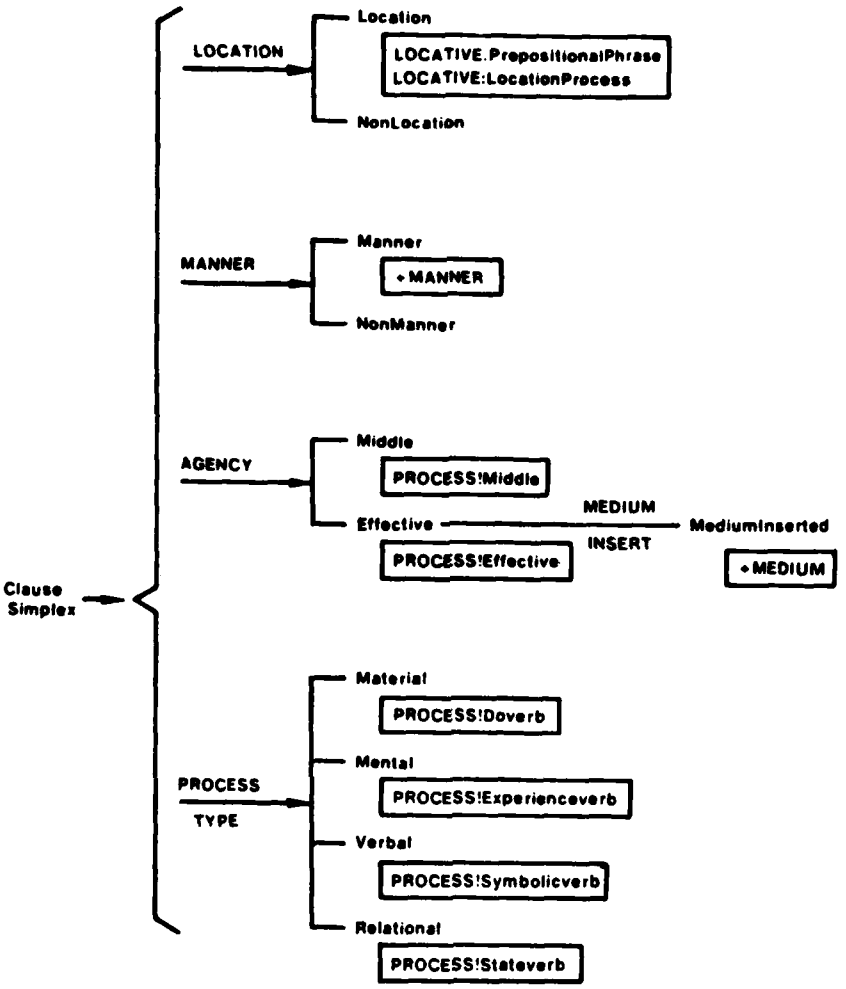


Figure 2-8: Transitivity I

2.4.3.2 MediumInsert

GRAMMAR: We now enter the Gate **MediumInsert**.

REALIZER: Insert MEDIUM.

NARRATOR: ***In addition to the systems of the grammar, we have entities called gates. A gate is best thought of as the entry condition part of a system, with only one output feature. This feature can play the same***

Table 2-4: Function Association Table, stage: (3)

HUBS	GRAMMATICAL FUNCTIONS
WREN-GAZEBO	ONUS
WREN-GAZEBO-STATEMENT	SPEECH-ACT
NOW	SPEAKING-TIME
GAZEBO-BUILDING	PROCESS
HISTORIC-TIME	EVENT-TIME
HISTORIC-TIME	RELEVANT-TIME
SIR-CHRISTOPHER	CAUSER
GREENWICH	LOCATION
MEDIOCRITY	MANNER

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material
```

Figure 2-9: Selection Expression, stage: (3)

role in the grammar in terms of realization, and be an input feature to other systems just as a feature in a system can. Even though only one output is possible, a gate can have a chooser that asks useful questions.

MediumInsert is a gate, but it does not have a chooser. It is present entirely to carry the realization operation we just saw, "Insert MEDIUM".

2.4.3.3 Location

GRAMMAR: We now enter the **Location** system. The choice options are NonLocation and Location.

CHOOSER: Is there knowledge of the location where GAZEBO-BUILDING (PROCESS) occurred?

ENVIRONMENT: It's known.

CHOOSER: What is the most salient location aspect of the knowledge represented by GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: Greenwich.

CHOOSER: I'll put the association of GREENWICH with LOCATION in Table 2-4.

CHOOSER: Is it preferable to express the knowledge represented by GREENWICH (LOCATION) or not?

ENVIRONMENT: Withhold it.

CHOOSER: Then I choose feature NonLocation (added to the selection expression in Figure 2-9).

CHOOSER: ... and I'm finished in this chooser.

2.4.3.4 Manner

GRAMMAR: We now enter the **Manner** system. The choice options are NonManner and Manner.

CHOOSER: Is there knowledge of the manner of performance or occurrence of GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: It's known.

CHOOSER: What is the most salient aspect of the manner of performance of GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: MEDIOCRITY.

CHOOSER: I'll put the association of MEDIOCRITY with MANNER in Table 2-4.

CHOOSER: Is it preferable to express the knowledge represented by MEDIOCRITY (MANNER) or not?

NARRATOR: ***Notice that the wording is familiar, from its use in the Location chooser. Whenever a particular wording of a question reappears, as in this case, the same underlying formal inquiry is being addressed to the environment.***

ENVIRONMENT: Withhold the expression of that knowledge.

CHOOSER: Then I choose feature NonManner (added to the selection expression in Figure 2-9).

CHOOSER: ... and I'm finished in this chooser.

2.4.3.5 ProcessType

GRAMMAR: We now enter the **ProcessType** system. The choice options are Material, Verbal, Mental, and Relational.

CHOOSER: Does the process GAZEBO-BUILDING (PROCESS) represent a static condition or state of being?

ENVIRONMENT: No, it doesn't represent a static condition or state of being.

CHOOSER: Does the process GAZEBO-BUILDING (PROCESS) represent symbolic communication of a kind which could have an addressee?

ENVIRONMENT: No, it isn't symbolic communication that could have an addressee.

CHOOSER: Is GAZEBO-BUILDING (PROCESS) a process of comprehension, recognition, belief, perception, deduction, remembering, evaluation or mental reaction?

ENVIRONMENT: No, it is not any of those.

CHOOSER: Then I choose feature Material (added to the selection expression in Figure 2-9).

CHOOSER: ... and I'm finished in this chooser.

REALIZER: Classify PROCESS Doverb.

2.4.4 Transitivity II

2.4.4.1 GoalInsertConflate

GRAMMAR: We now enter the **GoalInsertConflate** gate--see Figure 2-10--and activate the feature **GoalInsertedConflated**.

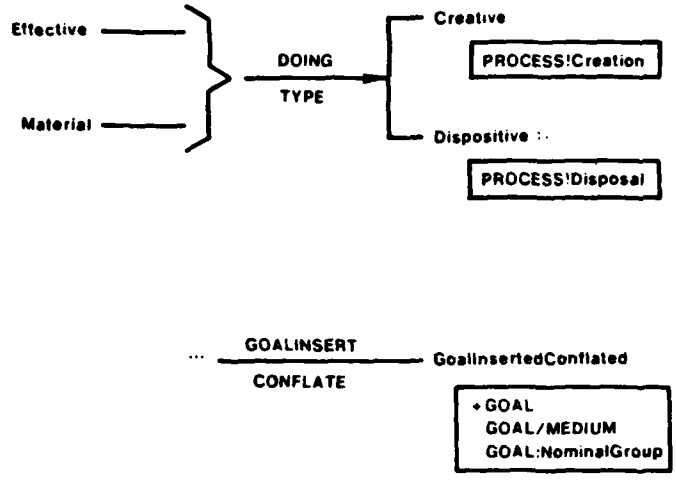


Figure 2-10: Transitivity II

REALIZER: Insert GOAL.

REALIZER: Conflate GOAL MEDIUM.

REALIZER: Preselect GOAL NominalGroup.

NARRATOR: **Preselect *associates a grammatical feature with a function. For each of the function bundles that carry preselections, the grammar will be reentered to develop the structure, and the preselected features will be chosen then. This gate, GoalInsertConflate, is here in order to invoke the realization that we have seen on a particular combination of features. Beyond this point we will not be showing entry of any gates, unless there is associated inquiry. This means that some realizations will appear mysteriously, and some of the entry conditions will be satisfied mysteriously.***

CHOOSER: What is the entity which GAZEBO-BUILDING (PROCESS) affects or is directed toward?

ENVIRONMENT: It's called THE-GAZEBO.

CHOOSER: I'll put the association of THE-GAZEBO with GOAL in Table 2-5.

NARRATOR: ***As we'll see, choosers know how to transfer a hub from one grammatical function to another. This is done here, copying the hub THE-GAZEBO from GOAL to MEDIUM.***

CHOOSER: I'll copy the hub THE-GAZEBO from GOAL to MEDIUM and put the association between THE-GAZEBO and MEDIUM in Table 2-5.

Table 2-5: Function Association Table, stage: (4)

HUBS	GRAMMATICAL FUNCTIONS
WREN-GAZEBO	ONUS
WREN-GAZEBO-STATEMENT	SPEECH-ACT
NOW	SPEAKING-TIME
GAZEBO-BUILDING	PROCESS
HISTORIC-TIME	EVENT-TIME
HISTORIC-TIME	RELEVANT-TIME
SIR-CHRISTOPHER	CAUSER
GREENWICH	LOCATION
MEDIOCRITY	MANNER
THE-GAZEBO	GOAL
THE-GAZEBO	MEDIUM
SIR-CHRISTOPHER	ACTOR
SIR-CHRISTOPHER	AGENT

CHOOSER: What is the entity which caused or causes GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: It is SIR-CHRISTOPHER.

CHOOSER: I'll put the association of SIR-CHRISTOPHER with ACTOR in Table 2-5.

CHOOSER: Is SIR-CHRISTOPHER (CAUSER) identical to SIR-CHRISTOPHER (ACTOR)?

ENVIRONMENT: They are the same.

NARRATOR: ***That question does not really contribute anything new. It's just an error check. The hub for CAUSER was arrived at independently in Section 2.4.3 in the Agency chooser.***

CHOOSER: Again we copy a hub from one grammatical function to another. In this case we copy the hub SIR-CHRISTOPHER from ACTOR to AGENT; the resulting new association is in Table 2-5.

CHOOSER: Then I activate feature GoalInsertedConflated, which is added to the selection expression in Figure 2-11

CHOOSER: ... and I have finished choosing.

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material;
GoalInsertedConflated, Creative
```

Figure 2-11: Selection Expression, stage: (4)

2.4.4.2 DoingType

GRAMMAR: We now enter the **DoingType** system. The choice options are Creative and Dispositive.

CHOOSER: Did THE-GAZEBO (GOAL) exist before the beginning of GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: It did not exist.

CHOOSER: OK, I choose feature Creative (added to the selection expression in Figure 2-11).

CHOOSER: ... and I'm finished in this chooser.

REALIZER: Classify PROCESS Creation.

2.4.5 Clause Voice

2.4.5.1 EffectiveVoice

GRAMMAR: We now enter the **EffectiveVoice** system-- see Figure 2-12. The choice options are operative (i.e., clause-active) and receptive (i.e.. clause-passive).

CHOOSER: Is it preferable to mention SIR-CHRISTOPHER (AGENT) as the agent of GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: Mention him.

CHOOSER: Is there a paragraph containing WREN-GAZEBO (ONUS) which has a theme?

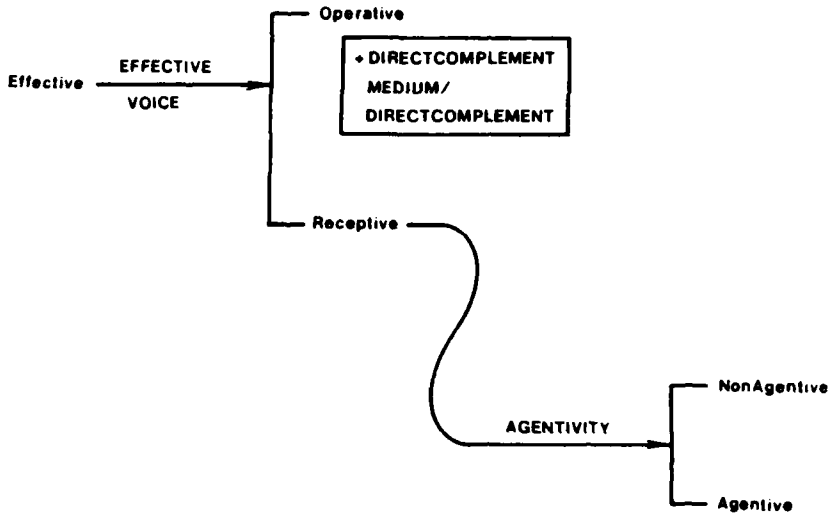


Figure 2-12: Clause voice

ENVIRONMENT: It exists.

CHOOSER: What is the theme of the paragraph containing WREN-GAZEBO (ONUS)?

ENVIRONMENT: It's called GREENWICH-REGION.

CHOOSER: I'll put the association of GREENWICH-REGION with PARAGRAPH-THEME in Table 2-6.

CHOOSER: What symbol represents the most salient chain of relationships in the reader's attention and knowledge between SIR-CHRISTOPHER (AGENT) and GREENWICH-REGION (PARAGRAPH-THEME)?

NARRATOR: ***This question is part of some reasoning about conceptual distance--represented by the notion of path--among the agent, the medium and the PARAGRAPH-THEME.***

ENVIRONMENT: It's called WREN-GREENWICH-REGION-PATH.

CHOOSER: I'll put the association of WREN-GREENWICH-REGION-PATH with AGENT-THEMATIC-PATH in Table 2-6.

CHOOSER: What symbol represents the most salient chain of relationships in the reader's attention and knowledge between THE-GAZEBO (MEDIUM) and GREENWICH-REGION (PARAGRAPH-THEME)?

ENVIRONMENT: It's called GAZEBO-GREENWICH-REGION-PATH.

CHOOSER: I'll put the association of GAZEBO-GREENWICH-REGION-PATH with MEDIUM-THEMATIC-PATH in Table 2-6.

Table 2-6: Function Association Table, stage: (5)

HUBS	GRAMMATICAL FUNCTIONS
WREN-GAZEBO	ONUS
WREN-GAZEBO-STATEMENT	SPEECH-ACT
NOW	SPEAKING-TIME
GAZEBO-BUILDING	PROCESS
HISTORIC-TIME	EVENT-TIME
HISTORIC-TIME	RELEVANT-TIME
SIR-CHRISTOPHER	CAUSER
GREENWICH	LOCATION
MEDIOCRITY	MANNER
THE-GAZEBO	GOAL
THE-GAZEBO	MEDIUM
SIR-CHRISTOPHER	ACTOR
SIR-CHRISTOPHER	AGENT
GREENWICH-REGION	PARAGRAPH-THEME
WREN-GREENWICH-REGION-PATH	AGENT-THEMATIC-PATH
GAZEBO-GREENWICH-REGION-PATH	MEDIUM-THEMATIC-PATH
THE-GAZEBO	SUBJECT

CHOOSER: Does the chain of relationships WREN-GREENWICH-REGION-PATH (AGENT-THEMATIC-PATH) contain the chain of relationships GAZEBO-GREENWICH-REGION-PATH (MEDIUM-THEMATIC-PATH) as a proper subpart?

ENVIRONMENT: Yes, it is a part of that one.

CHOOSER: Then I choose feature Receptive, which is added to the selection expression in Figure 2-13.

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material;
GoalInsertedConflated, Creative;
Receptive, Agentive
```

Figure 2-13: Selection Expression, stage: (5)

NARRATOR: ***This means that the chooser has now arrived at the decision that the***

MEDIUM is more closely related to the PARAGRAPH-THEME than the AGENT is.

CHOOSER: I now copy the hub THE-GAZEBO from MEDIUM to SUBJECT; the new association is in Table 2-6.

NARRATOR: ***We have learned a lot about practical development of the choosers from this particular chooser. It started as just a one-question chooser, asking about whether it was preferable to mention the agent. Later we added the interactions with paragraph theme, and some interactions you didn't see involving the previous clause. All of these changes were based on work by Sandra Thompson. Putting them into the chooser revealed some problems. Those were corrected, and changes from other sources have been added since.***

Developments like this one have shown that the choosers are definable with a wide range of levels of sophistication. It has also been easy to reconcile several people's work in the chooser development process.

CHOOSER: ... and I'm finished in this chooser.

2.4.5.2 Agentivity

GRAMMAR: We now enter the **Agentivity** system. The options are Agentive and NonAgentive.

CHOOSER: Is it preferable to mention SIR-CHRISTOPHER (AGENT) as the agent of GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: Mention him.

CHOOSER: Then I choose Agentive (added to the selection expression in Figure 2-13).

CHOOSER: ... and I'm finished choosing

NARRATOR: ***Several realizations will now appear mysteriously; the entry conditions to a number of gates have been satisfied.***

REALIZER: Insert AGENT.

REALIZER: Insert ACTOR.

REALIZER: Preselect ACTOR NominalGroup.

REALIZER: Conflate ACTOR AGENT.

REALIZER: Insert AGENTMARKER.

REALIZER: Lexify AGENTMARKER by.

REALIZER: Order AGENTMARKER AGENT.

REALIZER: Insert PASSIVE.

REALIZER: Classify PASSIVE Beaux.

REALIZER: Insert PASSPARTICIPLE.

REALIZER: Classify PASSPARTICIPLE En-participle.

2.4.6 Verbal Voice

GRAMMAR: We now enter the gate **LexverbPasspart**; see Figure 2-14.

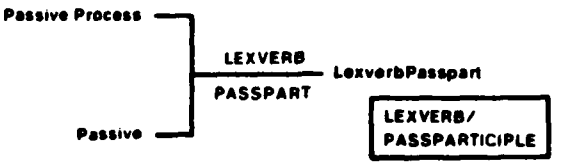


Figure 2-14: Verbal voice

CHOOSER: My next job is to determine the intersection of two term sets, which leads to a question that is much too long to read.

ENVIRONMENT: The answer is also much too long to read. It is a list of all the known en-participles.

CHOOSER: Then I activate feature LexverbPasspart, which is added to the selection expression; see Figure 2-16 below in Section 2.4.7.

REALIZER: Conflate LEXVERB PASSPARTICIPLE.

REALIZER: Conflate MEDIUM SUBJECT.

CHOOSER: ... and I'm finished in this chooser.

2.4.7 Mood I

2.4.7.1 Dependence

GRAMMAR: We now enter the **Dependence** system-- see Figure 2-15. The choice options are IndependentClause and DependentClause.

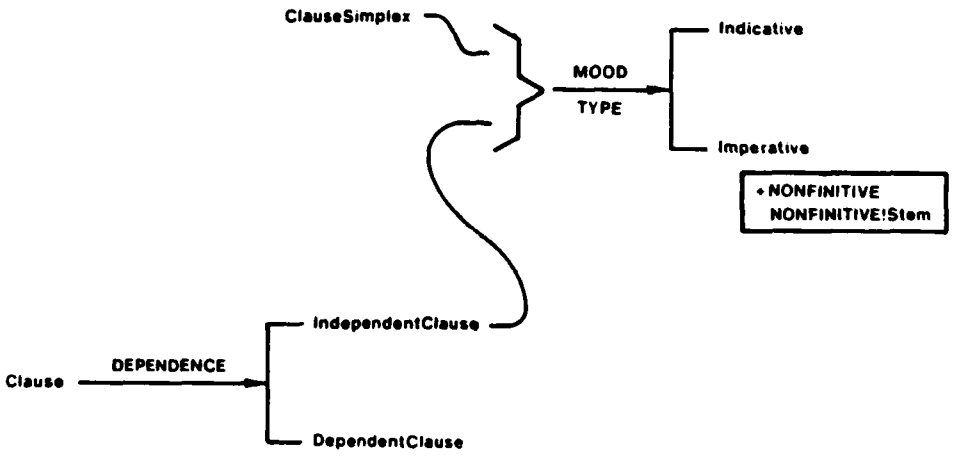


Figure 2-15: Mood I

CHOOSER: Does WREN-GAZEBO-STATEMENT (SPEECH-ACT) represent the entire speech act or is it part of a larger one?

ENVIRONMENT: It's the whole thing.

CHOOSER: Then I choose feature IndependentClause, which is added to the selection expression in Figure 2-16.

CHOOSER: ... and I'm finished in this chooser.

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material;
GoalInsertedConflated, Creative;
Receptive, Agentive;
IndependentClause, Indicative
```

Figure 2-16: Selection Expression, stage: (6)

2.4.7.2 MoodType

GRAMMAR: All right, we now enter the **MoodType** system. The choice options are Indicative and Imperative.

CHOOSER: Is the illocutionary point of the surface level speech act represented by WREN-GAZEBO-STATEMENT (SPEECH-ACT) a command, i.e., a request of an action by the hearer?

ENVIRONMENT: It's not intended to command.

CHOOSER: Then I choose feature Indicative (added to the selection expression in Figure 2-16).

CHOOSER: ... and I'm finished in this chooser.

2.4.8 Mood II

2.4.8.1 IndicativeType

GRAMMAR: Fine. We now enter the **IndicativeType** system--see Figure 2-17. The choice options are Declarative and Interrogative.

CHOOSER: Is the illocutionary point of the surface level speech act represented by WREN-GAZEBO-STATEMENT (SPEECH-ACT) to state?

ENVIRONMENT: Yes, it's intended to state.

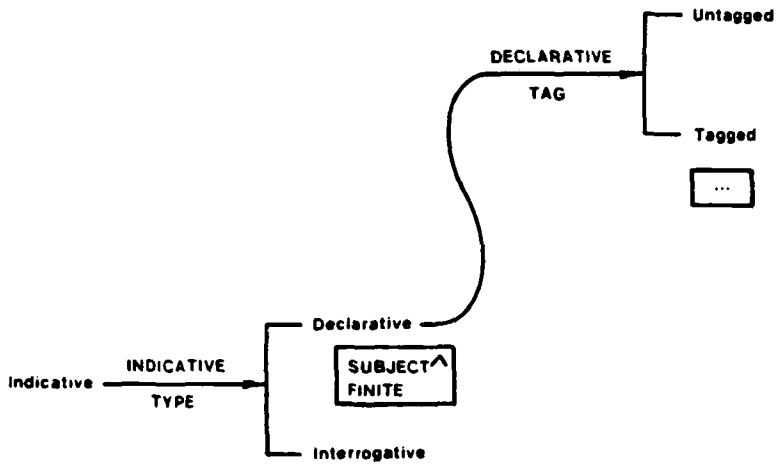


Figure 2-17: Mood II

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material;
GoalInsertedConflated, Creative;
Receptive, Agentive;
IndependentClause, Indicative;
Declarative, Untagged
```

Figure 2-18: Selection Expression, stage: (7)

CHOOSER: Then I choose feature Declarative, which is added to the selection expression in Figure 2-18.

CHOOSER: ... and I'm finished in this chooser.

REALIZER: Order SUBJECT FINITE.

2.4.8.2 DeclarativeTag

GRAMMAR: We now enter the **DeclarativeTag** system. The choice options are Untagged and Tagged.

CHOOSER: Should a secondary speech act be performed in conjunction with WREN-GAZEBO (ONUS)?

ENVIRONMENT: There is no need for a secondary speech act.

CHOOSER: Then I choose feature Untagged (added to the selection expression in Figure 2-18).

CHOOSER: ... and I'm finished in this chooser.

2.4.9 Thematization

GRAMMAR: We now enter the **ThemeMarkingDeclarative** system; see *Figure* 2-19. The choice options are UnMarkedDeclarativeTheme and MarkedDeclarativeTheme.

CHOOSER: What concept is specified as the theme of WREN-GAZEBO (ONUS)?

ENVIRONMENT: THE-GAZEBO is.

CHOOSER: I'll put the association of THE-GAZEBO with THEME in Table 2-7.

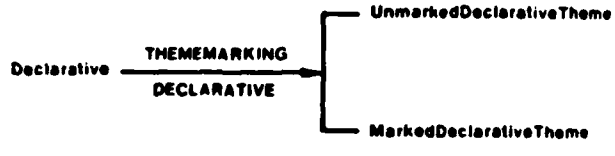


Figure 2-19: Thematization

Table 2-7: Function Association Table, stage: (6)

HUBS	GRAMMATICAL FUNCTIONS
WREN-GAZEBO	ONUS
WREN-GAZEBO-STATEMENT	SPEECH-ACT
NOW	SPEAKING-TIME
GAZEBO-BUILDING	PROCESS
HISTORIC-TIME	EVENT-TIME
HISTORIC-TIME	RELEVANT-TIME
SIR-CHRISTOPHER	CAUSER
GREENWICH	LOCATION
MEDIOCRITY	MANNER
THE-GAZEBO	GOAL
THE-GAZEBO	MEDIUM
SIR-CHRISTOPHER	ACTOR
SIR-CHRISTOPHER	AGENT
GREENWICH-REGION	PARAGRAPH-THEME
WREN-GREENWICH-REGION-PATH	AGENT-THEMATIC-PATH
GAZEBO-GREENWICH-REGION-PATH	MEDIUM-SUBJECT-PATH
THE-GAZEBO	SUBJECT
THE-GAZEBO	THEME

CHOOSER: As theme, is THE-GAZEBO (THEME) to be expressed in a marked way?

ENVIRONMENT: No, no special emphasis is needed.

CHOOSER: Then I choose feature UnmarkedDeclarativeTheme, which is added to the selection expression in Figure 2-20.

CHOOSER: ... and I'm finished in this chooser.

REALIZER: Insert MOOD.

REALIZER: Expand MOOD FINITE.

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material;
GoalInsertedConflated, Creative;
Receptive, Agentive;
IndependentClause, Indicative;
Declarative, Untagged;
UnmarkedDeclarativeTheme
```

Figure 2-20: Selection Expression, stage: (8)

2.4.10 Deicticity

2.4.10.1 Deicticity

GRAMMAR: We now enter the **Deicticity** system-- see Figure 2-21. The choice options are Temporal and Modal.

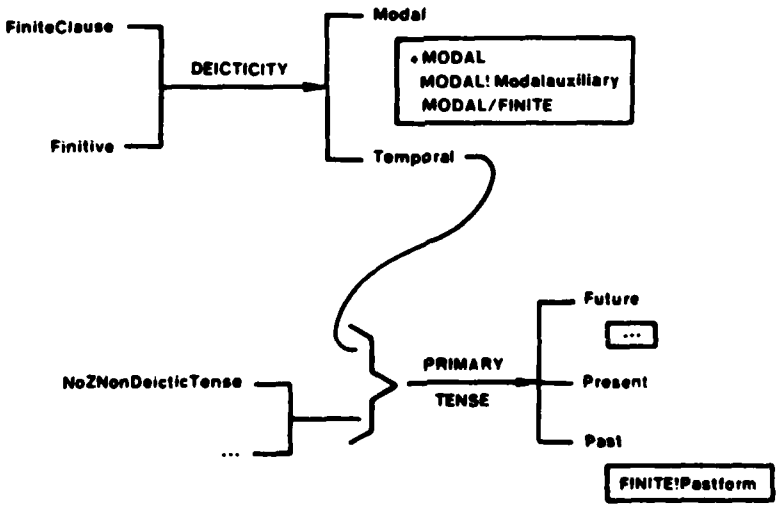


Figure 2-21: Deicticity

CHOOSER: Does GAZEBO-BUILDING (PROCESS) have a definite existence, possibly hypothetical, in space and time?

ENVIRONMENT: It has a definite existence.

CHOOSER: Then I choose feature Temporal, which is added to the selection expression in Figure 2-22.

CHOOSER: ... and I'm finished in this chooser.

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material;
GoalInsertedConflated, Creative;
IndependentClause, Indicative;
Declarative, Untagged;
UnmarkedDeclarativeTheme;
Temporal, Past
```

Figure 2-22: Selection Expression, stage: (9)

2.4.10.2 PrimaryTense

GRAMMAR: We now enter the **PrimaryTense** system. The choice options are Past, Present, and Future.

CHOOSER: Is GAZEBO-BUILDING (PROCESS) extensional, i.e., concrete, rather than intensional, i.e., generic?

ENVIRONMENT: It's extensional.

CHOOSER: Is the condition or process represented by GAZEBO-BUILDING (PROCESS) hypothetical?

ENVIRONMENT: It's real, not hypothetical.

CHOOSER: Does the moment or interval of time HISTORIC-TIME (RELEVANT-TIME) strictly precede the moment or interval NOW (SPEAKING-TIME)?

ENVIRONMENT: It precedes it.

CHOOSER: Then I choose feature Past (added to the selection expression in Figure 2-22).

CHOOSER: ... and I'm finished in this chooser.

REALIZER: Insert FINITE.

REALIZER: Classify FINITE Pastform.

REALIZER: Conflate FINITE PASSIVE.

2.4.11 Polarity

2.4.11.1 Polarity

GRAMMAR: We now enter the **Polarity** system-- see Figure 2-23. The choice options are Positive and Negative.

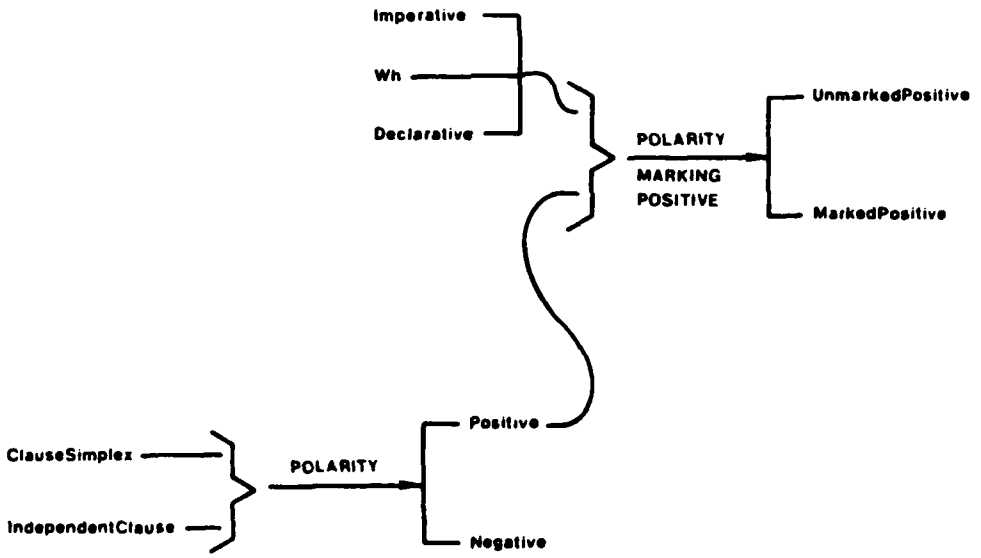


Figure 2-23: Polarity

CHOOSER: What concept represents the polarity of assertion or occurrence of GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: It's called GAZEBO-BUILDING-POLARITY.

CHOOSER: I'll put the association of GAZEBO-BUILDING-POLARITY with POLARITY in Table 2-8.[15]

---

[15]This table is the final version and appears in full in Table 2-9 below in Section 2.6.

Table 2-8: Function Association Table, stage: (7)

HUBS	GRAMMATICAL FUNCTIONS
...	
GAZEBO-BUILDING-POLARITY	POLARITY

CHOOSER: What is the value of the polarity of GAZEBO-BUILDING-POLARITY (POLARITY)?

ENVIRONMENT: It's a positive one.

CHOOSER: Then I choose feature Positive, which is added to the selection expression in Figure 2-24.

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material;
GoalInsertedConflated, Creative;
IndependentClause, Indicative;
Declarative, Untagged;
UnmarkedDeclarativeTheme;
Temporal, Past;
Positive, UnmarkedPositive
```

Figure 2-24: Selection Expression, stage: (10)

CHOOSER: ... and I'm finished in this chooser.

2.4.11.2 PolarityMarkingPositive

GRAMMAR: We now enter the **PolarityMarkingPositive** system. The choice options are UnmarkedPositive and MarkedPositive.

CHOOSER: Should GAZEBO-BUILDING-POLARITY (POLARITY) be given distinguished emphatic or contrastive status?

ENVIRONMENT: No special emphasis is needed.

CHOOSER: Then I choose feature UnmarkedPositive (added to the selection expression in Figure 2-24).

CHOOSER: ... and I'm finished in this chooser.

2.4.12 Subject Person

2.4.12.1 ThingSubject

GRAMMAR: We now enter the **ThingSubject** system--see Figure 2-25. The choice options are Conscious and NonConscious.

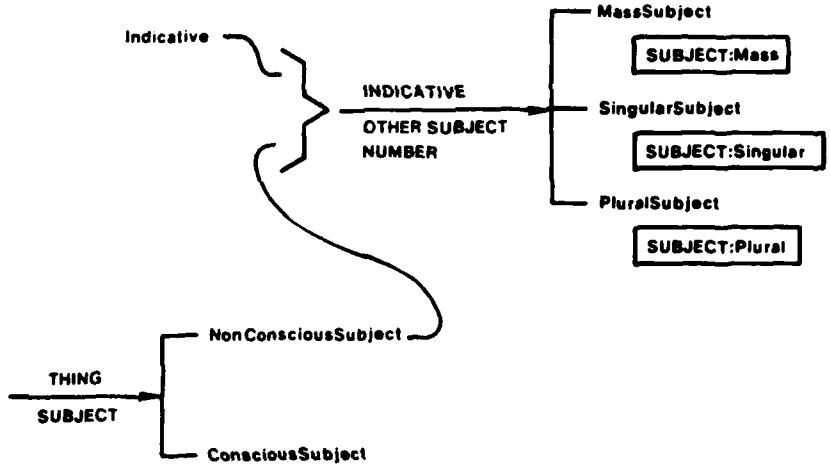


Figure 2-25: Subject person

CHOOSER: Is the individual or group represented by THE-GAZEBO (SUBJECT) conscious?

ENVIRONMENT: It's not conscious.

CHOOSER: Then I choose feature NonConsciousSubject, which is added to the selection expression in Figure 2-26.

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material;
GoalInsertedConflated, Creative;
IndependentClause, Indicative;
Declarative, Untagged;
UnmarkedDeclarativeTheme;
Temporal, Past;
Positive, UnmarkedPositive;
NonConsciousSubject, SingularSubject
```

Figure 2-26: Selection Expression, stage: (11)

CHOOSER: ... and I'm finished in this chooser.

NARRATOR: ***Now several gates that have been waiting do some realization.***

REALIZER: Insert SUBJECT.

REALIZER: Expand MOOD SUBJECT.

REALIZER: Insert TOPICAL.

REALIZER: Expand THEME TOPICAL.

REALIZER: OrderAtEnd TOPICAL.

REALIZER: OrderAtFront THEME.

REALIZER: Preselect SUBJECT Nominative.

REALIZER: Conflate TOPICAL SUBJECT.

2.4.12.2 IndicativeOtherSubjectNumber

GRAMMAR: We now enter the **IndicativeOtherSubjectNumber** system. The choice options are SingularSubject and PluralSubject.

CHOOSER: Is THE-GAZEBO (SUBJECT) inherently multiple, i.e., a set or collection of things, or unitary?

ENVIRONMENT: It's unitary.

CHOOSER: Then I choose feature SingularSubject (added to the selection expression in Figure 2-26).

CHOOSER: ... and I'm finished in this chooser.

REALIZER: Preselect SUBJECT Singular.

REALIZER: Classify FINITE Singular.

REALIZER: Classify FINITE Thirdperson.

2.4.13 LexicalVerb TermResolution

GRAMMAR: We now enter **LexicalVerbTermResolution**, a gate which has a chooser so that it can ask questions about the lexical verb.

CHOOSER: Again I must find the intersection of two term sets, and again it leads to a question too long to read.

ENVIRONMENT: The reply is a list of three en-participles, for build, create and construct.

NARRATOR: ***Notice that the result is the set of remaining terms in Figure 2-5 in boldface:*** **built, constructed, created.**

CHOOSER: Select one of these terms: (builten createden constructeden) (LEXVERB) to express GAZEBO-BUILDING (PROCESS)?

ENVIRONMENT: I select "built."

REALIZER: I therefore eliminate "created" and "constructed."

CHOOSER: Then I activate feature LexicalVerbTermResolution, which is added to the selection expression in Figure 2-27.

2.4.14 Final Realization

REALIZER: There are no more systems or gates which can be entered. I do the remaining ordering according to lists of ordered functions.

REALIZER: The resulting ordered set of function bundles is:

(TOPICAL SUBJECT GOAL MEDIUM)
(FINITE PASSIVE)
(PASSPARTICIPLE PREDICATOR LEXVERB PROCESS)
AGENTMARKER
(ACTOR AGENT)
  Realized as:
TOPICAL + SUBJECT + GOAL + MEDIUM was built by ACTOR + AGENT

At this point, the hub SIR-CHRISTOPHER is associated with the function ACTOR. I could realize this by clearing the Feature Association Table, associating ONUS with SIR-CHRISTOPHER, and entering the grammar with NominalGroup preselected. However, we won't show this. We could realize the SUBJECT similarly. Nigel handles both of these appropriately, but lack of space prevents showing the details.

```
Clauses, Clause, ClauseSimplex;
NoNonDeicticPresent, NoZNonDeicticTense;
Effective, MediumInserted, NonLocation,
NonManner, Material;
GoalInsertedConflated, Creative;
IndependentClause, Indicative;
Declarative, Untagged;
UnmarkedDeclarativeTheme;
Temporal, Past;
Positive, UnmarkedPositive;
NonConsciousSubject, SingularSubject;
LexicalVerbTermResolution
```

Figure 2-27: Selection Expression, stage: (12): final expression

2.5 RESULTS OF THE GENERATION PROCESS

We have now completed one cycle in the process of generating *This gazebo was built by Sir Christopher Wren*. As records of this process, we have accumulated a selection expression--shown above in Figure 2-27--which is a list of all the features chosen (in systems) or activated (in gates). During generation, feature realizations were applied to the features chosen by Realizer and he also specified ordering according to ordered lists of functions. The result is the function structure for the clause shown in Figure 2-28. Here, each column represents one functionally specified constituent, and is the result of the Conflation operation.

Throughout the generation process the Chooser has been keeping track of hub-to-function associations in the growing function association table. The final version of it is displayed in Table 2-9.

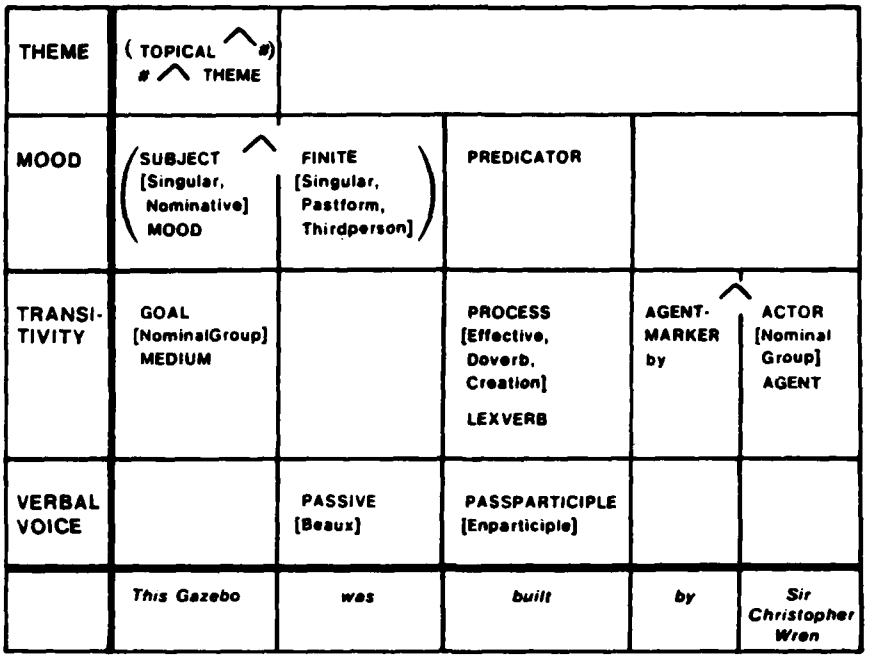


Figure 2-28: Clause structure

2.6 REVIEW

We have seen a detailed example of how text can be generated according to an intention and a plan. Although text generation is not a central concern for most of us, the methods and devices used can be applied in other ways.

1. The chooser framework can serve as a starting point in explorations of the design of a systemic semantics.
2. Choosers also serve as a way of describing the semantics of a given language; the choosers in Nigel constitute the beginnings of a semantics of grammatical choice for English.
3. The chooser and inquiry framework has proven itself very useful as a source of insights and preferences on the grammar itself. It has often been possible to choose between two different systemic representations of a particular grammatical phenomenon, based on the fact that one led to a neat collection of choosers and the other did not. Dependencies among choosers often point to meta-functional organization that is not directly evident from the wiring of the grammar itself.
4. We expect that the choosers will be useful in teaching English, since they give a basis, supported by some research, for explaining the uses of the various grammatical options of the language. Imagine using the EffectiveVoice chooser to help explain the use of the English passive.

Table 2-9: Function Association Table: complete state

HUBS	GRAMMATICAL FUNCTIONS
WREN-GAZEBO	ONUS
WREN-GAZEBO-STATEMENT	SPEECH-ACT
NOW	SPEAKING-TIME
GAZEBO-BUILDING	PROCESS
HISTORIC-TIME	EVENT-TIME
HISTORIC-TIME	RELEVANT-TIME
SIR-CHRISTOPHER	CAUSER
GREENWICH	LOCATION
MEDIOCRITY	MANNER
THE-GAZEBO	GOAL
THE-GAZEBO	MEDIUM
SIR-CHRISTOPHER	ACTOR
SIR-CHRISTOPHER	AGENT
GREENWICH-REGION	PARAGRAPH-THEME
WREN-GREENWICH-REGION-PATH	AGENT-THEMATIC-PATH
GAZEBO-GREENWICH-REGION-PATH	MEDIUM-THEMATIC-PATH
THE-GAZEBO	SUBJECT
THE-GAZEBO	THEME
GAZEBO-BUILDING-POLARITY	POLARITY

5. The choosers also seem useful for discourse studies, since they make it easier to show how particular grammatical options are responsive to discourse information.
6. The set of inquiries of the grammar could be useful in language comparison, in the service both of contrastive analysis and of typological studies. Since each system of the grammar is syntactically motivated, the set of inquiries represents the semantics of grammatical encoding of English. This factors its semantics into an explicit grammatical part and the rest. We expect this factoring to be useful in semantic studies.

We hope that this informal demonstration has stimulated your imagination. There are two other papers in this conference which deal with the chooser and inquiry framework and the Nigel grammar. They describe things at a more abstract level.

REFERENCE

[Halliday 70] Halliday, M.A.K., "Functional diversity in language, as seen from a consideration of modality and mood in English," *Foundations of Language* 6.3, 1970, 327-351.

3. THE SYSTEMIC FRAMEWORK IN TEXT GENERATION: NIGEL

Christian M. I. M. Matthiessen

ABSTRACT

A large, systemic grammar has been implemented computationally and a semantic stratum has been developed for it. The grammar and its semantics, called Nigel, are intended as a component in a text generation system. This paper describes the systemic framework as it has been worked out for Nigel in the context of the text generation task: Many properties of the systemic framework are justified by the text generation task and this will be indicated. In addition, the task has led to clarifications, revisions and additions to the framework. This paper is independent of Chapters 1 and 2, but can usefully be read in the context of those papers as the last in the series.[16]

3.1 INTRODUCTION

This paper reports on work on systemic grammar and semantics carried out by Mann, Halliday, and Matthiessen, building on previous work by Halliday ([Halliday 61] and onwards, particularly [Halliday 69] for the form of the grammar). In addition to design and theory, the result of the research is a systemic grammar and a corresponding semantics, both of which have been implemented computationally; this component is called Nigel.[17] The present discussion is primarily about the grammar (and semantic) framework and machinery, not about English. It is in some sense "meta-linguistics." Throughout the discussion I will indicate how the design of the grammar relates to the text generation task.

3.1.1 Grammar: Tasks, Framework and Form

There is a growing awareness in linguistics--certainly not new in systemic linguistics--that the structure and organization of language are influenced by its functions. The same holds true at the metalevel--also an observation with a long history in systemic linguistics, see [Halliday 64]; the grammatical framework and the notation should match the task of the grammar.

Any new task for a grammar is a challenge and test for the framework. The task provides us with a working environment in which to develop the framework and a test bed in which to check whether we achieve the goals defined by the task. The systemic grammar to be presented, Nigel, has

[16]This paper draws on the work carried out by W. Mann, M.A.K. Halliday, and C. Matthiessen in the research on Nigel. I am very grateful to Bill Mann for many comments on drafts of this paper and to Sandy Thompson, David Webber, Jim Melancon, and to participants in the 9th International Systemic Workshop for comments on earlier oral or written versions of the paper. Needless to say, I am solely responsible for all errors.

[17]The grammar is still developing and expanding, but it is by far the largest computational systemic grammar and integrates descriptions of English grammar by Halliday spanning approximately twenty years. The organization of the grammar as presented here reflects the current organization of Nigel. It does not necessarily reflect Halliday's views on what the organization of a systemic grammar and semantics should ideally be like given a particular task.

been developed for a task that is very central in language use, viz. text generation. The task has a number of consequences for the grammar which will be presented after a characterization of text generation.

3.1.2 The Character of the Text Generation Task

The text generation task can be understood in terms of what a system that deals with that task, a text generation system, must do to generate text.

In a given situation, a need to generate a text may arise. This need will lead to a specification of the goals to be achieved by the generated text (its intent, purpose, functional tenor, or rhetorical mode) and of values for the variables of the communicative situation.

To achieve these goals, a text planning component creates a text plan. The plan terminates in clusters of messages or propositions that can be handled by the grammar: Nigel takes over and starts grammatical encoding. Nigel has to be sensitive to global principles of organization in the text plan as they influence terminal propositional clusters. This is needed to handle, for example, theme selection and conjunction. Nigel must also be sensitive to the locally specified goals and "propositional content." Thus, Nigel's role is defined by the rest of the text generation system, which will be called the Environment. In Nigel, we will distinguish the semantic component, called the Choosers, from the grammatical component, which includes the network representation, called Grammar, and a realizational part, called Realizer.

3.1.3 Demands on Grammar from the Text Generation Task

The text generation task leads to demands for a grammar that

(I) in terms of framework and notation:

1. formalizes grammar in such a way that it can serve the rest of the text generation system. For instance, it should be possible to relate abstractions in the grammar to abstractions in the environment and to control the grammar so as to achieve the desired effects.
2. states the grammar fully explicitly. All aspects of the organization of the grammar have to be "filled in."

(II) in terms of the contents of the grammar:

1. is conceived of as a whole and not as a collection of mini-grammars for disconnected areas of grammar.
2. has good overall coverage of communicatively important areas of English grammar rather than just those which test the formalism.

In this presentation of Nigel, I will concentrate on demands (I.1-2); the demands relating to the coverage of the grammar have been dealt with elsewhere [Matthiessen 82]. For the demands under (II), it can just be noted that work done in the systemic framework supports the coverage demands

very well. The demands of (I) have been consequential for the framework in a number of ways. Basically, (I.1) has brought out many aspects of the systemic framework that fit the text generation task extremely well. In addition, there have been formal clarifications and new designs.

3.1.4 Organization of this Chapter

It is useful to discuss the organization and processes of Nigel in terms of four dimensions, all of which have a tradition in systemic linguistics. These dimensions are **stratification**: semantics and grammar; **axis**: paradigmatic and syntagmatic; **cycle**:[18] an arbitrary cycle *n* and the next one *n + 1*, and, finally, **potentiality**: potential and actual. As this list illustrates, each dimension defines two constructs (semantics and grammar, paradigmatic and syntagmatic, etc.). The organization within each construct will be called intra-organization (intra-stratal, intra-axis, etc.); the relations that relate the two constructs of one dimension will be called inter-relations (inter-stratal, inter-axis, etc.) or realizations. The present discussion will concentrate on the first three dimensions, i.e., on stratification, axis, and cycle.

The presentation will be guided by the generation of one particular clause-structure (which can represent for example *This gazebo was built by Sir Christopher Wren*).[19] Each aspect of Nigel will be introduced by a step in the generation of this clause-structure. Section 3.3 underlines the value of separating the paradigmatic axis from the syntagmatic axis in a text generation grammar and brings in the dimension of stratification as well to show how an explicit paradigmatic organization can be used. Section 3.4 continues the discussion of the axis dimension, but focuses on inter-axis relations (feature-to-function realization). Section 3.5 introduces an additional dimension, the cycle, and shows how the problem of inter-cyclic relations has been solved in Nigel. Section 3.6 ends the generation of the example clause and presents the organization of the syntagmatic axis in Nigel, i.e., of the function structure. There follows, in Section 3.7, a summary of the three dimensions introduced.

The example used to illustrate the grammar appears in italics throughout the discussion. The components (Grammar, Chooser, Environment, and Realizer) are as described in Section 3.1.2 above. If read continuously, the illustrative italicized parts constitute a summary of the generation of one clause, presented in detail in the generation drama (Chapter 2).

3.2 GRAMMAR AND SEMANTICS: PARADIGMATIC AND SYNTAGMATIC STRATIFICATION

This section deals with the issue of relating the Nigel grammar to the rest of the text generation system. This has to be addressed by the systemic principle of stratification, which has been explored in the work of Nigel through the design of a semantic stratum--the so-called chooser framework developed for Nigel--and through a design of the interaction of grammar and semantics. It is argued that two basic systemic functional principles have been instrumental: the separation of the

---

[18] The term and the notion will be explained below.

[19] For a detailed transcript of how Nigel generates this clause, see Chapter 2, in this volume. The illustrations used for this paper have been excerpted from that transcript and edited to fit as examples here.

representation of the paradigmatic axis of grammatical organization from the syntagmatic one and the functional representation of syntagmatic organization.

3.2.1 Separating Paradigmatic from Syntagmatic

In this section we will show how grammar and semantics (the choosers of the grammar) work together and point to how the systemic view and design of grammar make it possible to create a systemic grammar for a text generation system.

We start with the grammar offering a choice.

GRAMMAR: We now enter the **Rank** *system. The choice options are* Clauses *and* GroupsPhrases.

As all systemic grammars from the mid 60s onwards, Nigel is built on the principles that (i) the two axes of organization, the paradigmatic axis and the syntagmatic one, are given separate representations, and (ii) the syntagmatic representation is derived from the paradigmatic one. The primitive of the paradigmatic organization is the **feature**, as for example Clauses in the example above. In Nigel, as in all systemic grammars, features are organized into systems, which, in turn, form a system network.

When the two basic dimensions of linguistic organization are separated, i.e., when we separate paradigmatic and syntagmatic organization, and take the paradigmatic dimension as primary (see [Halliday 66])--to be deep grammar (to use an essentially non-systemic metaphor)--we get a grammar where the paradigmatic organization faces semantics. Or, to put it in another way, the paradigmatic organization is the way into grammar from semantics. The result is a clear separation of grammar as a choice potential (paradigmatic axis) and the means by which the choices are realized (syntagmatic axis). This separation gives us a handle on a systemic grammar as a text generation grammar: It is the choice points in grammar that have to be controlled, and these are stated explicitly as systems. As the characterization of demand (I.1) above indicates, this property of systemic grammars is highly desirable given the text generation task. Let us now move on to how this control can be designed.

3.2.2 Stratification: The Control of Grammar

The Grammar has offered a choice, a statement of its local resources. The next interactant is the Chooser.

CHOOSER: Does WREN-GAZEBO (ONUS) have its own illocutionary force?

ENVIRONMENT: Yes, it has an illocutionary force.

CHOOSER: Then I choose feature Clauses.

We are at the Rank system in the grammar and the task is now to choose between the two features Clauses and GroupsPhrases. In other words, we need to control the system. This is not a task for the grammar itself, but for the next level (stratum). The device used is a **chooser**.

One clearly defined role for semantics[20] is to be an **interlevel** or **interface**. Halliday (in [Halliday 61]) writes that the meaning of an item "is its *relation* to extratextual features." Semantics is, then, an interlevel between grammar and what is outside language; it is "the strategy that is available for entering the language system" ([Halliday 73], p. 64).

A chooser works as an interface in the way suggested by the example above. It is assigned to a particular system, i.e., it is the chooser of a particular system, and asks the environment questions that will supply it with the information needed to choose appropriately. The Rank chooser (as all choosers) faces both the grammar (through the choice or selection in the grammar) and what is outside the linguistic (Nigel) system (called "environment" in the dialogue). The chooser tests for the distinctions that have to obtain in the environment and when a particular distinction or a particular set of distinctions obtain, the chooser **realizes** these distinctions that are non-grammatical through the grammar by choosing a feature. We will meet other types of realization as we proceed; this type is **inter-stratal realization.**[21]

What is outside language is a highly complex system, as for example Halliday [Halliday 78] shows. In the work on text generation systems, it is useful to see what is outside language as the **environment**[22] of semantics and grammar with initially only two parts--a knowledge base and text-plans (from a text-planner).

3.2.3 Associations between Concepts and Functions

A chooser relates to the grammar by choosing a feature in a system based on information from the environment. This is the paradigmatic interaction with the grammar. In addition, a chooser may create an association between a grammatical function and a concept from the environment. This is the syntagmatic interaction.

The locus in the environment which the choosers can go to to trace information relevant to their questions is called WREN-GAZEBO. Early on in the generation, once it has been established that we're generating a simple clause, the following exchange occurs:

CHOOSER: What concept is the process aspect of WREN-GAZEBO?

ENVIRONMENT: It's called GAZEBO-BUILDING.

CHOOSER: I associate the concept GAZEBO-BUILDING with the grammatical function PROCESS.

---

[20] There are currently various kinds of semantics that differ according to what they set out to do. The question is not so much "which semantics is the right one?" but rather "what are the tasks we need to recognize for semantics?"

[21] "Realization" is the general term for a move along one of the three dimensions stratification, axis, and cycle. We will distinguish inter-stratal realization, inter-axis realization (feature-to-function realization), and inter-cyclic realization (preselection).

[22] There is an obvious correspondence between the term *environment* and the term *context* from Firthian linguistics. However, it is better to use *environment* so as not to make or suggest any implicit assumptions about the nature of what is outside Nigel.

Here the chooser creates an association between the GAZEBO-BUILDING concept from the environment and the PROCESS function from the grammar. In Nigel, as in many systemic functional grammars, the primitive of syntagmatic representation is the grammatical function. So grammatical functions rather than classes carry associations. A constituent may be characterized by more than one function.[23] Consequently, a constituent may have more than associations, for example, a constituent may simultaneously be TOPICAL (i.e., topical theme), SUBJECT, and ACTOR.

The type of multi-functional grammar Halliday has developed over the years enables us to manipulate function-to-concept associations in a more sophisticated way than if we simply had one association per constituent. Each constituent may have three functions (at least) in the clause--a transitivity function, a mood function, and a theme function--each one of which may give rise to the concept association carried by this bundle of functions.

If we represent the inter-stratal relations introduced here by "Choose" (paradigmatic) and "Associate" (syntagmatic), we can summarize the discussion simply, as illustrated in Figure 3-1.

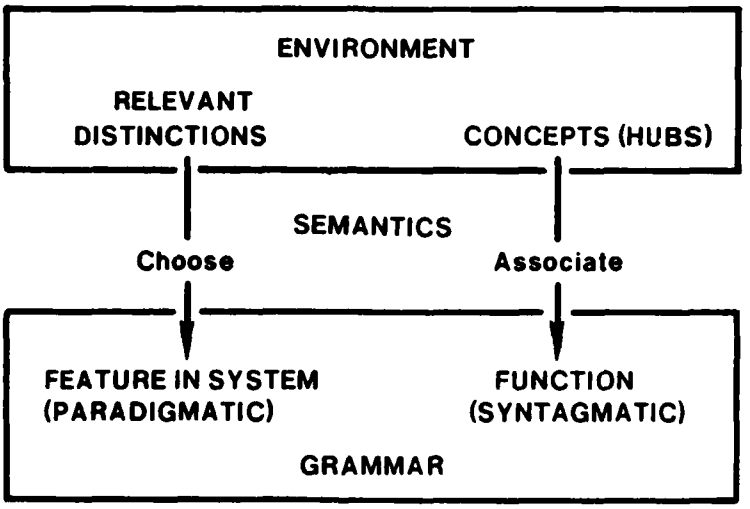


Figure 3-1: Semantics as an active interlevel

When we generate we shunt back and forth between strata, between semantics and grammar. This is the general strategy for all inter-dimension work, as shown below.

3.3 ASPECTS OF PARADIGMATIC ORGANIZATION

The paradigmatic organization of grammar is the best explored aspect of systemic grammar. This section confines itself to a brief presentation of the gate and a few other notions that are not "universal" in systemic grammar. It is also noted how, for example, a disjunctive entry condition is interpreted.

---

[23] Generally speaking, this is a result of Halliday's meta-functional hypothesis: each metafunction may contribute one or more micro-functions as the characterization of the constituent.

3.3.1 Paradigmatic Organization as Single Network

The first system entered from Nigel was Rank. The entire system network can be reached from this system. Halliday [Halliday 69] has described the grammar of English as one huge network and this is what the fragment represented in the Nigel grammar is.

This means that the notion of rank has been represented as an early system in the grammar; in this respect the Nigel grammar is like Hudson's grammar of English complex clauses in [Hudson 71]. There are a number of consequences of treating the grammar formally as a single network.

3.3.2 Dimensions of the Network: Simultaneity and Dependency

In the system Rank the feature Clauses was chosen. This feature is the input to another system which the grammar now offers:

GRAMMAR: We now enter the **ClauseClass** *system. The choice options are Clause and Clausette.*

After appropriate questions to the Environment, the ClauseClass chooser selects Clause, which is the input to yet another system:

GRAMMAR: We now enter the **ClauseComplexity** *system. The choice options are ClauseSimplex and ClauseComplex.*

Again after interaction with the Environment, a choice can be made and the ClauseComplexity chooser selects ClauseSimplex, which is the entry condition to (among others) the systems Agency and Manner. In other words, Agency and Manner are simultaneous in the network, both having ClauseSimplex as their entry condition:

GRAMMAR: We now enter the **Agency** *system. The choice options are Middle and Effective.*

And, after some chooser activity:

GRAMMAR: We now enter the **Manner** *system. The choice options are NonManner and Manner.*

The move from left to right in a system network is a move in dependency: from a feature or a collection of features to a system whose entry condition they satisfy. In the present example, the move from the system ClauseClass to the system ClauseComplexity is a move in dependency. Each step such as this one increases the **delicacy**[24] (or "depth").[25] Since the grammar is a single network

---

[24] Delicacy, as a technical term, here only applies to the paradigmatic axis--in contrast to, e.g., [Halliday 61] where elements of structure were specified to various degrees of delicacy

[25] In spatial metaphor terms, the network has breadth (or width) and depth. However, since "depth" has already been used as a technical term with another meaning--see, e.g., [Huddleston 65]--and since we already have the term "delicacy," I shall use "delicacy" and "simultaneity" instead of "depth" and "breadth" here.

one, the first two steps in delicacy have a special status; the first step is **rank** and the second step is **(primary) classes** at each rank--as shown in Figure 3-2.

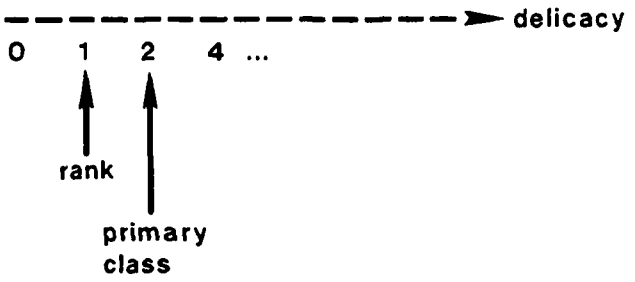


Figure 3-2: Rank and class in Nigel

The system ClauseClass was entered above. It represents the two primary classes of clauses, Clause and Clausette (major and minor clauses). If we had chosen GroupsPhrases instead of Clauses in the Rank system, we would have reached the system that specifies primary class at group/phrase rank--see Figure 3-3.

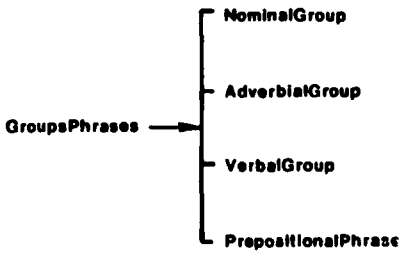


Figure 3-3: Primary class at group/phrase rank

So. rank and primary class are aspects of delicacy in Nigel. Delicacy is the measure of the horizontal dimension of the network, i.e., of its dependency dimension. The other dimension, the vertical dimension, will simply be called **simultaneity**. The systems Agency and Manner (in the example above) are simultaneous systems in the network.

3.3.3 The Points of Connection: Systems and Gates

Up to now, we have met only systems in the generation of "This gazebo was built by Sir Christopher Wren." However, once the chooser has chosen the feature Effective, the entry condition to a gate has been satisfied:

GRAMMAR: We now enter the Gate **MediumInsert**.

Here the Grammar does not present a choice option. There is only one output feature.

The gate is a device that is widely used in Nigel, but which has not been standard in systemic grammars before. (For an earlier discussion of various network devices, see a discussion by Fawcett [Fawcett].) Systems and gates both have features as inputs and outputs: they are composed of left-facing (upward, in Stratificational terminology) and right-facing (downward) disjunctions or conjunctions--or, if they are not feature complexes, their inputs and outputs are single features: see Figure 3-4.

COMPLEXITY / DIRECTION	Single feature:	Feature complex	
		Conjunction	Disjunction
Right:output	GATE	(Not defined)	SYSTEM
Left:input	All Possible for both GATE and SYSTEM		

Figure 3-4: System and gate

Feature complexes as inputs can have, for example, a conjunction with a disjunction as one of the conjuncts. In other words, there can be more than one layer.

The disjunction of the output of a system is exclusive: only one of the output features (terms in the system) may be chosen. A disjunction in the input is not exclusive, however. If the input to a system is a disjuntion of the features Extent and TopicalExtent, both of the disjuncts may be satisfied. However, this only counts as one entry of the system: a feature can only be chosen once in a pass through the grammar. For instance, we may have a specification of the temporal extent (i.e., duration) of the process, either for experiential reasons or for thematic reasons--or for both. So the entry condition of the system specifying the subtypes of extent in English has a disjunction of Extent and TopicalExtent as its input.

3.3.4 Network Conventions not Defined in Nigel

A small number of conventions have sometimes been used by systemicists to indicate various restrictions on the selection of feature combinations (symbols include daggers and stars).[26] These conventions have not been defined in Nigel--they do not in fact lend themselves to simple formalizations--but they can be viewed as shorthand for more explicit wiring in the network in Nigel.

Another network convention that has not been formalized in Nigel, but which remains informal with respect to the notation used in Nigel is the "loop" of recursive systems (typically found in the logical component). Henrici [Henrici 81] observes that "linear recursion is rather more troublesome" than recursion through embedding (which is achieved through preselection in Nigel) and concludes that "much work still remains to be done"; his remarks are still highly relevant.

---

[26] Cf. also Hudson's feature addition rules in Daughter Dependency Grammar [Hudson 76].

3.4 FROM PARADIGMATIC TO SYNTAGMATIC: FEATURE-TO-FUNCTION REALIZATION

Feature-to-function realization has not been as much explored in systemic linguistics as system networks and the nature of syntagmatic representation. This section presents the realization operators that have been specified for Nigel.

3.4.1 The Step from Paradigmatic to Syntagmatic

One of the systems reachable from the feature ClauseSimplex is ProcessType. The grammar and the chooser work together as before and the chooser makes a choice:

CHOOSER: Then I choose feature Material ... and I'm finished in this chooser.

REALIZER: Classify PROCESS Doverb.

It was shown in the previous section how the separation of the representation of the paradigmatic axis and the syntagmatic axis fits well with demands created by the text generation task. Once we have given the two axes separate representations, however, we need to relate them very explicitly. In other words, when we provide for separate intra-axis representations, we must also provide an inter-axis representation. In Nigel, this job is handled by feature-to-function realization statements.[27] Given a certain feature or combination of features, these statements, when activated, insert functions into the syntagmatic representation, i.e., into the function structure, and carry out other operations. For each type of realization statement (like insertion) there is a realization operator (like Insert). So, to include feature-to-function realization statements as inter-axis relations into our diagrammatic overview, we can expand the lower box of Figure 3-1 (leaving out the non-grammatical parts above the lower box) into Figure 3-5.

Figure 3-5: Inter-axis relations

3.4.2 Types of Inter-axis Realization Operators

The full set of inter-axis realization operators is given below in Figure 3-6. **Insert** specifies the presence of a function in the function structure being built. The other realization operators specify the syntagmatic relations a function can enter into, i.e., a constituency relation (**Expand**), a simultaneity relation (**Conflate**), and ordering relations (**Partition** and **Order, OrderAtEnd, OrderAtFront**).

---

[27] Since the paradigmatic organization is primary, the syntagmatic one is derived from it through realization statements in generation.

Insert and *Conflate* have direct equivalents in [Halliday 69]; *Partition* and *Order* correspond to what was called *concatenate* there. *Expand* is Halliday's name for the operator that specifies constituency. For example, both SUBJECT and FINITE can be specified as constituents of MOOD through the Expand operator: (Expand MOOD SUBJECT) and (Expand MOOD FINITE).[28]

OPERATOR	SYMBOL	EXAMPLE
(Insert X)	+X	+SUBJECT
(Conflate X Y)	X/Y	SUBJECT/AGENT
(Expand X Y)	X(Y)	MOOD(FINITE)
(Order X Y)	X↑Y	SUBJECT↑FINITE
(Partition X Y)	X|Y	PROCESS|MANNER
(OrderAtEnd X)	X↑#	TOPICAL↑#
(OrderAtFront X)	#↑X	#↑TEXTUAL

Figure 3-6: Realization operators: symbols

The final function clause-structure of the example being generated is the result of the application of most of the realization operators. In Figure 3-11 below, functions in vertical columns have been conflated. Expansion is indicated by a subscripted parenthesis.

Partition and *Order* both specify orderings of functions, taking a pair of functions as arguments. This first operator just specifies that one function precedes another. For example. (Partition SUBJECT FINITE) allows for the possibility of an intervening adjunct,[29] e.g., a conjunctive:

SUBJECT CONJUNCTIVE FINITE.

The second ordering operator is Partition plus adjacency: (Order SUBJECT FINITE) means that SUBJECT precedes FINITE and that the two functions are adjacent. (Order TO INFINITIVE) would mean that an infinitive cannot be split from its marker.

The distinction between the two ordering operators thus makes it possible to express the

---

[28]The expand operator only specifies constituency within one cycle (see below for this notion); it is used rather sparingly. Inter-cyclic preselection is brought about indirectly through preselection (also discussed below in Section 3.5).

[29]The operator corresponds to Martin Kay's SUBJECT ... FINITE notation.

observation that in a structure there are certain places or slots where elements, e.g., adjuncts, may intervene and that in other places they are less likely to.

Realizations are stated in the system network in connection with the features they realize--see the sample below in Figure 3-7--rather than in a separate list.[30]

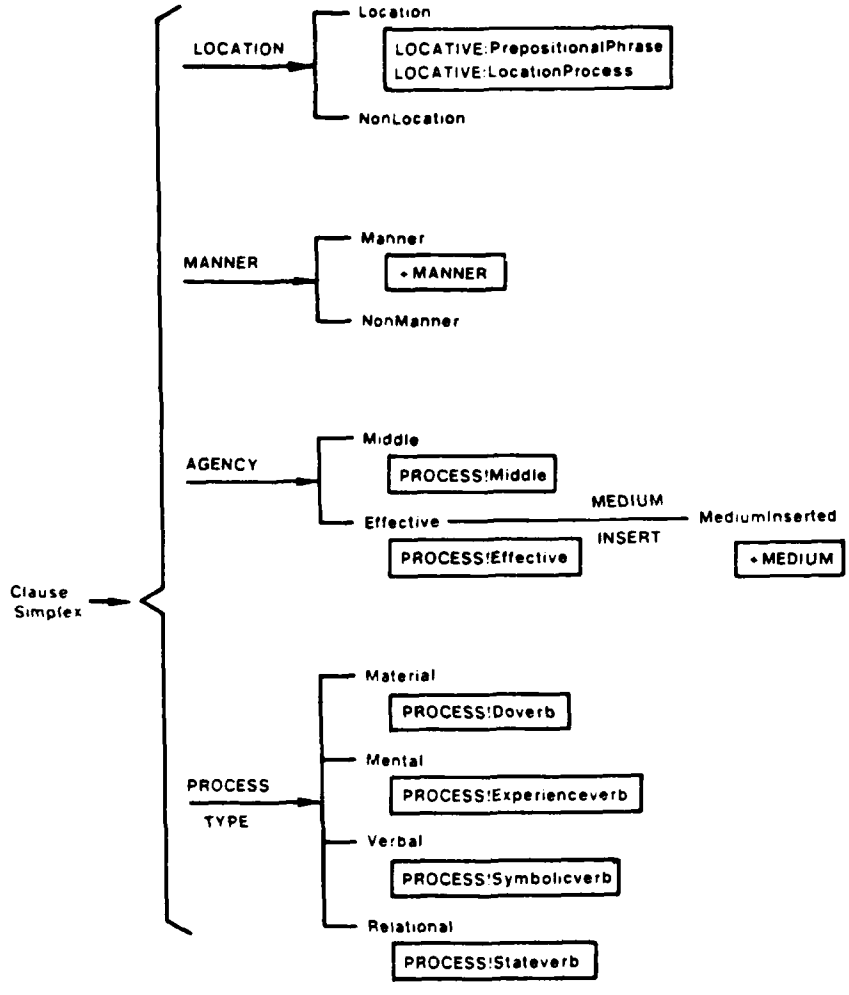


Figure 3-7: Fragment of network with realizations

[30] The separate list display is used for example in [Hudson 71], in [Davey 79], and in [Fawcett 80]

3.5 PARADIGMATIC TO PARADIGMATIC: INTER-CYCLIC REALIZATION

So far we have taken the generation of our clause as far as the paradigmatic representation and realization statements (as summarized in Figure 3-5 above) can take us. We have not seen the completion of the clause-structure yet, but that is within the capacity of the grammar as presented in the preceding sections (except for some additional ordering information that will be introduced below). It is comparable to the grammar illustrated by Halliday in [Halliday 69] and the notation defined by Henrici [Henrici 81]. However, we have neither exhausted the grammar nor reached the lexicon. This section will present how the grammar is re-entered or the lexicon entered to extend beyond one "feature-to-function" cycle. It follows from demand 1.2 in Section 3.1.3 that there has to be a fully explicit specification of how the grammar can be re-entered and how the lexicon can be reached. First, let me make clear what is meant by *cycle* in the present context.

3.5.1 The Notion of Cycle

Huddleston [Huddleston 65] defines the **depth** of an item in a structure as "the layer at which the item under consideration occurs" and says that it can be "evaluated quite simply, by counting the number of nodes between it and the top of the tree." Given this definition of depth, we can go on to say that the depth of an item is the syntagmatic (structural) reflection of the number of times we have **cycled** through the grammar to generate the item. One cycle can be defined as one complete move from the paradigmatic axis to the syntagmatic axis. Each cycle begins with an entry of the system network and ends with a structure fragment. We will now illustrate how Nigel moves from one cycle to another.

3.5.2 Preparing for Another Cycle: Preselection

In the previous examples, the Realizer has carried out feature-to-function realization, i.e., specified the presence of functions and how they are related. In the course of generation, the Realizer also deals with another type of realization (inter-cyclic realization):

REALIZER: (Preselect ACTOR NominalGroup).

REALIZER: (Preselect GOAL NominalGroup).

For every functionally characterized constituent there is the possibility either (i) of re-entering the system network to make further choices to determine its internal structure or (ii) of going to the lexicon.

All the feature-to-function operators capture the general systemic notion of preselection, the term being used for one of them. The feature associated with a function is either grammatical and the operator is **Preselect** or lexical and the operator is **Classify** or **Outclassify**.

Preselect associates a grammatical feature with a grammatical function, as a realization of one or more grammatical features. For example, in *This gazebo was built by Sir Christopher Wren*, the SUBJECT is determined to be singular in a clause system. This leads to a preselection of the SUBJECT to be a singular nominal group: (Preselect SUBJECT Singular). In this way, Preselect

defines an inter-cyclic relation from the cycle in which we make a pass through the clause part of the network to the cycle in which we make a pass through the nominal group part of the network to specify the SUBJECT.

Any feature in the system network has one or more **paths** leading to it, i.e., a set of choices through which it can be reached. As long as there is only one path leading to the feature, it can be preselected and its path computed through redundancy, so-called **path augmentation**. In other words, on a unique path only the most delicate feature need be preselected. When we return to the system network in the next **cycle**, the preselection and the path augmentation make it clear which part of the network to enter.

Classify is like Preselect, the only difference being that the operator associates a lexical feature with a function and not a grammatical one. The lexical features used in Nigel have not been systemicized, so path augmentation is only used for preselected grammatical features and not for classified lexical ones.

Outclassify specifies a lexical feature that must *not* be present in the feature set of a lexical item realizing a particular function bundle.[31] For example, (Classify FINITE Reduced) means that the lexical item realizing FINITE is required to have Reduced in its feature set. In contrast, (Outclassify FINITE Reduced) means that the lexical item is required not to have Reduced in its feature set.

There is an additional operator that makes contact with the lexicon, viz. **Lexify**. The two previous operators. Classify and Outclassify, specify lexical items paradigmatically through features that characterize them. Each Classify or Outclassify specifies a set of lexical items. In contrast, Lexify specifies a unique lexical item.[32] (Lexify FINITE will) does not mean that FINITE should be realized by one of the members of the set characterized as "will" items, but that FINITE should be realized by the item *will* itself.[33]

To sum up, preselection serves an inter-role in Nigel just as feature-to-function realization does. In fact, we can define the general notion **realization** as an inter-relation; it is inter-stratal, inter-axis, or inter-cycle. Choose and associate are the inter-stratal relations, the various feature-to-function realizations like Insert and Expand are the inter-axis relations, and Preselection is the inter-cycle relation (see Figure 3-8).

3.5.3 The Role of the Gate in Realization

In general, a realization is only stated once, even if there are different paradigmatic conditions under which it applies. This is accomplished by means of the **gate**. For example, there are various conditions under which SUBJECT may be inserted and, instead of repeating (Insert SUBJECT) each time, the following gate is used (see Figure 3-9).

---

[31] A function bundle or fundle is the result of applying realization operators to single functions. In other words, it is a micro-functionally characterized constituent.

[32] Actually, to be more precise, Lexify specifies a unique form of a lexical item.

[33] Lexify could be characterized as specifying a form of a lexical item syntagmatically. However, for ease of reference, Preselect, Classify, Outclassify, and Lexify will all be treated as instances of the preselection relation between paradigmatic representations at adjacent cycles. For a discussion of a related issue, see Halliday [Halliday 63].

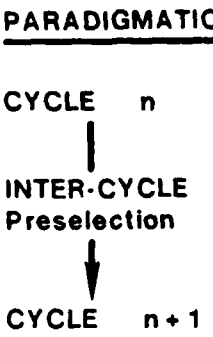


Figure 3-8: Preselection, the inter-cyclic realization

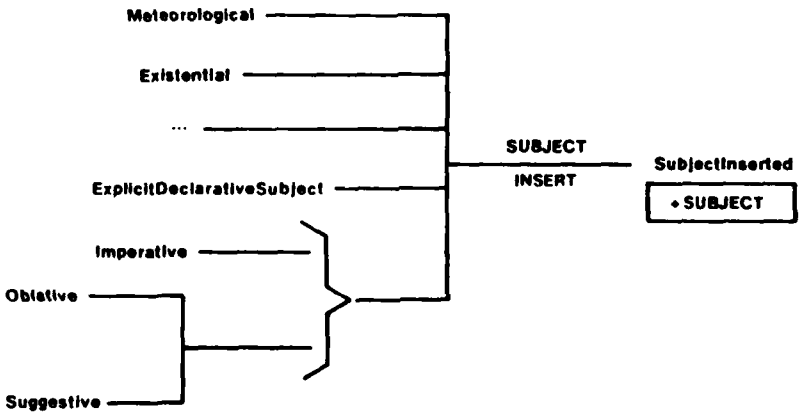


Figure 3-9: Subject insertion by gate

The gate can also be used for so-called conditional realizations. For instance, we may say "Given feature m, (Insert F), if feature n is also present." Formally, this is equivalent to saying "Given features m and n, (Insert F)" and we can use a gate to do the job for us, without using an additional mechanism; see Figure 3-10 for this gate.

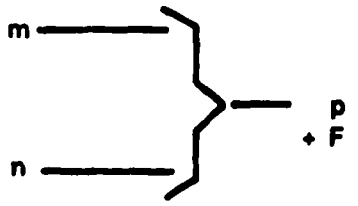


Figure 3-10: Gate used for conditional realization

As presented above, the part of the network with realization statements (Figure 3-7) is just the display sequence of realizations; it still leaves open different possible application sequences.

3.5.4 The Status of Realization in Systemic Grammar

As already mentioned in the introduction to this section, in systemic linguistics generally realization statement conventions are not in as stable a state as the system network notation. The set of realization operators used in Nigel has been sufficient so far and can be used as the basis for further explorations, for example, of the best way of arriving at the right sequence of constituents.

3.5.5 Function Realization: An Inter-cyclic Relation not Used in Nigel

The type of inter-cyclic realization that has been presented most fully in the literature is **function realization** (see e.g., [Hudson 71]). Function realization statements specify features (at cycle n + 1) given a certain function (at cycle n). For instance, Hudson gives the realization [ing-form] (i.e., a feature) for the function GERUND.

Function realization statements are moves along two dimensions: from cycle n to cycle n + 1 (cycle) and from function to feature (axis). In contrast, preselections stay within the paradigmatic axis and only move from one cycle to another.

Often the two realization strategies are fully equivalent. However, there is a drawback with function realization: Whatever information we already have stated paradigmatically at cycle n (for example, the [ing-form] environment) has to be recoded (via feature-to-function realization) syntagmatically so that function realization can apply and carry the information into the next cycle (via function realization). In the worst case, we have to create specific functions whose purpose is only to carry the information about a particular form in a conjugation or a particular lexical item. This may contribute to the proliferation of grammatical functions that has been noted for the variant of systemic grammar making use of function realization (cf. remarks in [Davey 79]).

In Nigel, function realization (as defined and used by Hudson [Hudson 71]) was first implemented, but was later eliminated, since it has proved possible to do without this type of realization entirely.

3.6 SYNTAGMATIC REPRESENTATION: FUNCTION STRUCTURE

We have illustrated nearly all the steps that lead to a fully specified and ordered clause-structure. Repeated choosing takes us through the system network. When all feature-to-function realizations have been applied, the result is the full function clause-structure of "This gazebo was built by Sir Christopher Wren." The structure without features specified by the preselection operators is presented in Figure 3-11. (The subscripted parentheses represent expansion and the vertical alignment represents conflation.)

The function structure above is not a fragment: it is the full clause-structure. There is, however, some ordering specification that the realization statements do not give. The so-called function order lists are not "feature realization," but can be viewed as a final set of **ordering constraints** on function structures.

(TOPICAL) THEME				
(SUBJECT MOOD	FINITE)			
GOAL MEDIUM		PROCESS	AGENTMARKER	ACTOR AGENT
	PASSIVE	PASSIVE-PARTICIPLE		

Figure 3-11: Function structure of clause

3.6.1 Function Order Lists

Although making ordering explicit (cf. demand I.2 in Section 3.1.3) in Nigel required quite a bit of work, the form of the ordering constraints is very simple; they are simply list of functions--for example,

```
(MANNER TIME LOCATION)
```

This says that, among the three circumstantials in the list, MANNER precedes TIME, which precedes LOCATION. The ordering is like "Partition": no adjacency is required.

These lists will have an effect only if the functional constituents have not been given a conflicting ordering by feature realization, i.e., by Order or Partition. For instance, it would be possible to state the ordering of SUBJECT and FINITE only for yes/no-interrogatives (forgetting about wh-interrogatives), i.e., Interrogative > (Partition FINITE SUBJECT), and have (SUBJECT FINITE) as one of the function order lists. If the clause generated is not Interrogative, no ordering of SUBJECT and FINITE has been specified and the list applies. However, when the clause is Interrogative, an ordering has been specified and the list (SUBJECT FINITE) does not apply.

3.6.2 Syntagmatic Rules not Used in Nigel: Structure Building Rules

As was stated above, the function structures generated by Nigel are fully specified by the feature-to-function realizations except for the ordering specifications by the function order lists. In this, Nigel differs from, for example, Hudson's grammar in [Hudson 71] which is more complicated.

Hudson has a category of **structure building rules**. These include rules similar to Nigel's function order lists, but in addition there are **compatibility rules**, **addition rules**, and **conflation rules** in Hudson's grammar. It has not been necessary to include any of these rules in Nigel.

3.7 SUMMARY: ORGANIZATION OF THE NIGEL GRAMMAR

The overall organization and design of the Nigel grammar can be brought out by looking at the two axes of organization, the paradigmatic axis and the syntagmatic axis, and their inter-relations within a cycle and across two cycles. As a result, we can bring Figures 3-1, 3-5, and 3-8 above together here into one figure that summarizes the organization of Nigel diagrammatically, see Figure 3-12.

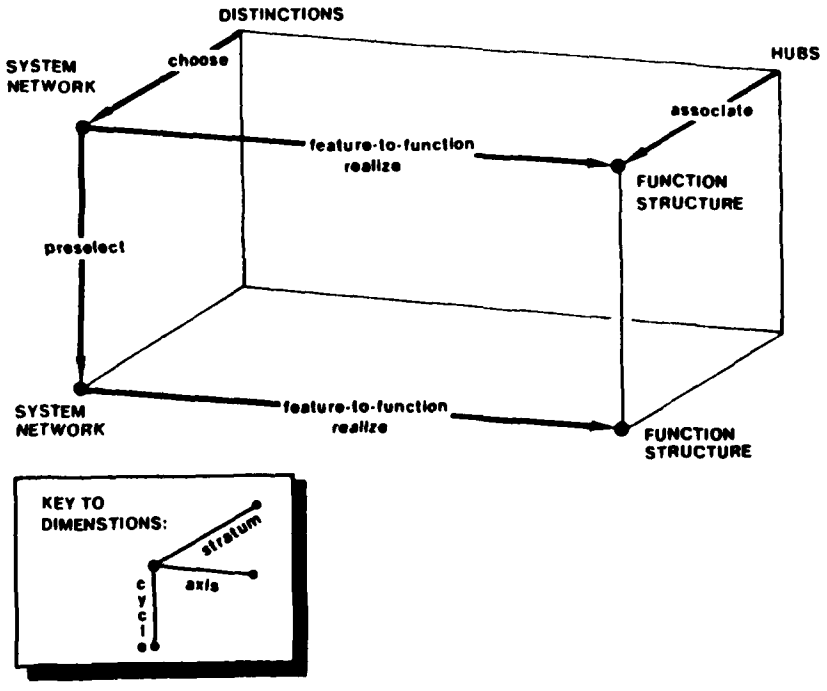


Figure 3-12: Inter- and intra-organization of Nigel

I will summarize Nigel in two steps: (i) intra-organization (network and structure); (ii) inter-relations.

3.7.1 Intra-axis and Intra-cycle Organization

Within a given cycle *n*, the system network represents the paradigmatic organization. The function structure represents the syntagmatic organization within a given cycle. These two, the network of systems and the linear structure of functions, are the intra-axis and intra-cycle modes of organization of language. All the other aspects of the Nigel grammar are inter-relations, either inter-axis or inter-cycle.

The primitives of the network and the structure are the feature and the function respectively. Both enter into intra-dimension relations that define the other constructs--systems, function bundles,

etc. Figure 3-13 tabulates the primitives, the relations the primitives enter into, and the resultant complexes of primitives, for both axes. The relations are different for the two axes[34] and this is how we can justify having features distinct from functions in Nigel: the operations performed on features are different from those performed on functions. For example, functions are conflated, but features are not.

There is a difference between the two axes that is not suggested in Figure 3-13 but which is basic to systemic grammars. The relations that define paradigmatic complexes (systems and gates) are stated at that level. In contrast, the relations that define syntagmatic complexes are not stated as syntagmatic rules (as happens in most current generative grammars) but as feature realizations.

	PARADIGMATIC	SYNTAGMATIC
Primitive	feature	function
Relations among primitives	Conjunction, disjunction, dependency	Conflation, expansion, ordering
Complexes	Systems, gates	Function bundles,[35] function structures

Figure 3-13: Paradigmatic and syntagmatic intra-organization

3.7.2 Inter-relations

There are theoretically more **inter-relations** than are actually used in the Nigel grammar at present. "Paradigmatic (at cycle) *n* to syntagmatic (at cycle) *n*" and "paradigmatic at *n* to paradigmatic at *n* + 1," i.e., feature-to-function realization and preselection respectively, are the only ones used at present.

In Figure 3-14, a complement to Figure 3-12, inter-relations that have been used in other systemic grammars and many inter-relations that could theoretically be used are shown (indicated by broken lines).

The lines (numbered for reference) are to be interpreted as follows.

---

[34] In this, systemic grammar differs from stratificational grammar where the same relations are used for both axes

[35] A fundle or a function bundle is the result of applying realization operators to single functions. In other words, it is a micro-functionally characterized constituent.

Broken line [1] means that a chooser would choose features at cycle n + 1 in addition to choosing features at cycle n. At present, the effect of this is only accomplished indirectly, through preselection. So, for instance, number agreement between SUBJECT and FINITE is accomplished by selecting for subject number in the clause and then preselecting the SUBJECT and classifying the FINITE through inter-cyclic (grammatical) realization according to the selection the chooser made in the clause.

In fact, inter-cyclic communication is done only by preselection in the grammar. As was already mentioned in Section 3.5, function realization (line [2]) is not necessary in Nigel. In addition, there is no inter-cyclic feature-to-function realization (line [3]): if a choice in the clause has effects for the structure of a group, this is handled by preselecting an appropriate feature for the group which, in its turn, will then be realized by an operation of feature-to-function realization.

Line [4] represents inter-cyclic feature-to-function realization where the direction is from one cycle to the preceding cycle. A possible example would be a selection in a group, the realization of which was an ordering of one of the functions of the group structure with respect to a function in the clause-structure.

Line [5], finally, represents statements that would include functions from say both clause-structure and the nominal group structure realizing the subject constituent of the clause. For example, it is possible to imagine function default ordering lists taking on a role of that kind.

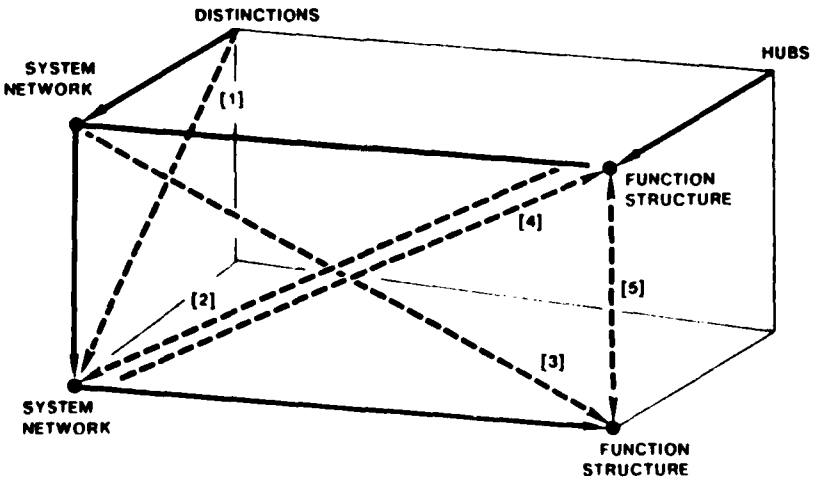


Figure 3-14: Theoretically possible inter-relations

The principles that constrain the Nigel grammar at present can be stated as follows:

1. Within a given cycle, paradigmatic is primary to syntagmatic.

2. Inter-relations never take more than one step at a time: they are either between the two axes keeping within one cycle, or between two adjacent cycles within the same axis--in fact, only within the paradigmatic axis.[36]

Note that the second constraint also states that there is for example no "unbounded" preselection from say cycle *n* to cycle *n* + 5. The diagram defines what Nigel can do [ability] within the present grammar framework [theoretical possibility]. It may, of course, turn out in future work either that Nigel ought to be able to do more or that the framework itself should be changed. The present set of inter-relations is listed in Figure 3-15.

[A] SEMANTIC (n) TO GRAMMATICAL (n)

Choose
Associate

[B] PARADIGMATIC (n) TO SYNTAGMATIC (n)

Insert
Expand
Conflate
Order, OrderAtEnd, OrderAtFront, Partition

[C] PARADIGMATIC (n) TO PARADIGMATIC (n + 1)

Preselect
Classify, Outclassify; Lexify

Figure 3-15: Summary of realization types

3.7.3 Another Dimension: Potentiality

Figure 3-12 basically represents the grammar as potential (in Halliday's sense of the term). Any particular example that we generate (such as *This gazebo was built by Sir Christopher Wren*) represents an actual instance of the potential; the generation process itself is an actualization (instantiation) of the potential; cf. [Halliday 73].

The diagram in the figure allows for more than one way of doing actualization; more than one "flow chart" can be derived from it. For instance, the entire system network can be actualized before

---

[36] As pointed out above, constituency can be defined as the syntagmatic relation between two adjacent cycles. However, there is no need for special statements to introduce it. The desired effect is achieved by preselection.

any structure is built or structure can be built (through feature realization) in tandem with the actualization of the system network. Issues such as this tend to be more prominent in an implemented grammar such as Nigel's, than in a grammar that has not been implemented and is used primarily for description. They are part of the demands of explicitness stemming from the text generation task.

Space constraints prevent me from discussing actualization issues here.

3.8 CONCLUSION

This chapter has presented a "fourth generation" Systemic Grammar that is a continuation of the work reported in [Halliday 69], the Nigel grammar.

The Nigel grammar has been developed for the task of text generation. This task provides both a test for the systemic framework and a research context in which aspects of systemic grammar that have been given less attention than for example the system network have been and can be further explored.

Two major design principles of systemic functional grammars as presented by Halliday in [Halliday 69] are central to the text generation task. The principles are (i) the separation of the paradigmatic and the syntagmatic organization, giving the former an independent representation, and (ii) a functional approach to grammar with for example structures stated in terms of functions.

The demand for explicitness inherent in the text generation task has given us reason to specify the inter-relations (realizations) in grammar and semantics in detail; the computational implementation has provided an opportunity to test these.

REFERENCES

[Davey 79] Davey, A., *Discourse Production,* Edinburgh University Press, Edinburgh, 1979.

[Fawcett 80] Fawcett, R. P., *Exeter Linguistic Studies.* Volume 3: *Cognitive Linguistics and Social Interaction,* Julius Groos Verlag Heidelberg and Exeter University, 1980.

[Fawcett] Fawcett, R. P., System networks, codes and knowledge of the universe: a cognitive perspective on the relationship between language and culture. To be published in M. A. K. Halliday, S. M. Lamb, and A. Makkai (eds.) *Semiotics of Culture and Language.*

[Halliday 61] Halliday, M. A. K., "Categories of the Theory of Grammar," *Word* 17, 1961.

[Halliday 63] Halliday, M. A. K., "Class in relation to the axes of chain and choice in language," *Linguistics* 2, 1963, 5-15.

[Halliday 64] Halliday, M. A. K., "Syntax and the consumer," in C. I. J. M. Stuart (ed.), *Report of the Fifteenth Annual (First International) Round Table Meeting on Linguistics and Language Study,* Georgetown University Press, 1964.

[Halliday 66] Halliday, M.A.K., "Some notes on 'deep' grammar," *Journal of Linguistics* 2.1, 1966, 57-67.

[Halliday 69] Halliday, M.A.K.. "Options and functions in the English clause." *Brno Studies in English* 8, 1969, 81-88.

[Halliday 73] Halliday, M. A. K., *Explorations in the Functions of Language*, Edward Arnold, 1973.

[Halliday 78] Halliday, M. A. K., *Language as Social Semiotic*, University Park Press, Baltimore, 1978.

[Henrici 81] Henrici, A., "Some notes on the systemic generation of a paradigm of the English clause." in M. A. K. Halliday and J. Martin (eds.), *Readings in Systemic Linguistics*, Batsford, London, 1981.

[Huddleston 65] Huddleston, R., "Rank and depth." *Language* 41, 1965, 574-86.

[Hudson 71] Hudson, R. A., *North-Holland Linguistic Series.* Volume 4: *English Complex Sentences*, North-Holland, London and Amsterdam, 1971.

[Hudson 76] Hudson, R. A., *Arguments for a Non-Transformational Grammar*, University of Chicago Press, Chicago, 1976.

[Matthiessen 82] Matthiessen, C. M. I. M., *The Syntactic Coverage of a Text Production Grammar*, USC/Information Sciences Institute, Technical Report, 1982.

other operations. For each type of realization statement (like insertion) there is a realization operator (like Insert). So, to include feature-to-function realization statements as inter-axis relations into our diagrammatic overview, we can expand the lower box of Figure 3-1 (leaving out the non-grammatical parts above the lower box) into Figure 3-5.

Figure 3-5: Inter-axis relations

3.4.2 Types of Inter-axis Realization Operators

The full set of inter-axis realization operators is given below in Figure 3-6. **Insert** specifies the presence of a function in the function structure being built. The other realization operators specify the syntagmatic relations a function can enter into, i.e., a constituency relation (**Expand**), a simultaneity relation (**Conflate**), and ordering relations (**Partition** and **Order**, **OrderAtEnd**, **OrderAtFront**).

---

[27] Since the paradigmatic organization is primary, the syntagmatic one is derived from it through realization statements in generation.